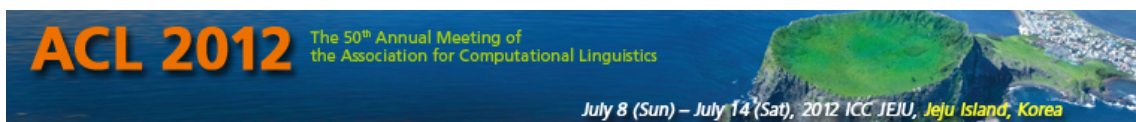


# 50th Annual Meeting of the Association for Computational Linguistics



## Proceedings of the Conference

Volume 1: Long Papers

July 8 - 14, 2012

Jeju Island, Korea

PLATINUM SPONSOR



GOLD SPONSORS



SILVER SPONSORS



BRONZE SPONSORS





SUPPORTERS



SPONSOR FOR BEST PAPER AWARD

**IBM Research**

©2012 The Association for Computational Linguistics

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](mailto:acl@aclweb.org)

ISBN 978-1-937284-24-4 (Volume 1: Long Papers)  
ISBN 978-1-937284-25-1 (Volume 2: Short Papers)

## Preface: General Chair

Welcome to Jeju Island — where ACL makes a return to Asia!

As General Chair, I am indeed honored to pen the first words of ACL 2012 proceedings. In the past year, research in computational linguistics has continued to thrive across Asia and all over the world. On this occasion, I share with you the excitement of our community as we gather again at our annual meeting. On behalf of the organizing team, it is my great pleasure to welcome you to Jeju Island and ACL 2012.

In 2012, ACL turns 50. I feel privileged to chair the conference that marks such an important milestone for our community. We have prepared special programs to commemorate the 50th anniversary, including ‘Rediscovering 50 Years of Discovery’, a main conference workshop chaired by **Rafael Banchs** with a program on ‘the People, the Contents, and the Anthology’, which recollects some of the great moments in ACL history, and ‘ACL 50th Anniversary Lectures’ by **Mark Johnson**, **Aravind K. Joshi** and a Lifetime Achievement Award Recipient.

A large number of people have worked hard to bring this annual meeting to fruition. It has been an unforgettable experience for everyone involved. My deepest thanks go to the authors, reviewers, volunteers, participants, and all members and chairs of the organizing committees. It is your participation that makes a difference.

Program Chairs, **Chin-Yew Lin** and **Miles Osborne**, deserve our gratitude for putting an immense amount of work to ensure that each of the 940 submissions was taken care of. They put together a superb technical program like nobody else. Publication Chairs, **Maggie Li** and **Michael White**, extended the publishing tools to take care of every detail and compiled all the books within an impossible schedule. Tutorial Chair, **Michael Strube**, put together six tutorials that you can never miss. Workshop Chairs, **Massimo Poesio** and **Satoshi Sekine**, working with their EACL and NAACL counterparts, selected 11 quality workshops, many of which are new editions in their popular workshop series. Demo Chair, **Min Zhang**, started a novel review process and selected 29 quality system demos. Faculty Advisors, **Kentaro Inui**, **Greg Kondrak**, and **Yang Liu**, and Student Chairs, **Jackie Cheung**, **Jun Hatori**, **Carlos Henriquez** and **Ann Irvine**, assembled an excellent program for the Student Research Workshop with 12 accepted papers. Mentoring Chair, **Joyce Chai**, coordinated the mentorship of 13 papers. Publicity Chairs, **Jung-jae Kim** and **Youngjoong Ko**, developed the website, newsletters, and conference handbook that kept us updated all the time. Exhibition Chair, **Byeongchang Kim**, coordinated more than 10 exhibitors with a strong industry presence. All the events are now brought to us on Jeju Island by the Local Arrangements Chairs, **Gary Lee** and **Jong Park**, and their team. I can never thank them enough for all the preparations they have made to host us in such a spectacular place!

I would like to express my gratitude and appreciation to **Kevin Knight**, Chair of the ACL Conference Coordination Committee, **Dragomir Radev**, ACL Secretary, and **Priscilla Rasmussen**, ACL Business Manager, for their advice and guidance throughout the process.

The financial sponsors generously supported ACL 2012 in a meaningful way despite a challenging

economic outlook. We are honored to have Baidu as the Platinum Sponsor, Elsevier and Google as Gold Sponsors, Microsoft, KAIST and SK as Silver Sponsors, 7 Bronze Sponsors, and 3 Supporters. The Donald and Betty Walker Student Scholarship Fund and Asian Federation of Natural Language Processing have supported our student travel grants. The sponsorship program was made possible by the ACL sponsorship committee: **Eiichiro Sumita, Haifeng Wang, Michael Gamon, Patrick Pantel, Massimiliano Ciaramita, and Idan Szpektor.**

Finally, I do hope that you have an enjoyable and productive time on Jeju Island, and that you will leave with fond memories of ACL's 50th Anniversary. With my best wishes for a successful conference!

Haizhou Li  
ACL 2012 General Chair  
July 2012

## Preface: Programme Committee Co-Chairs

This year we received 571 valid long paper submissions and 369 short paper submissions. 19% of the long papers and 20% of the short papers were accepted. As usual, some are presented orally and some as posters. Taking unigram counts from accepted long paper titles, and ignoring function words, the most popular word were:

entity 5  
evaluation 5  
hierarchical 5  
information 5  
joint 5  
syntactic 5  
topic 5  
discriminative 6  
lexical 6  
statistical 6  
chinese 7  
dependency 7  
machine 8  
modeling 8  
models 8  
language 10  
word 10  
parsing 11  
model 12  
learning 14  
translation 15

Some areas have grown over time and some have diminished. The most popular area for submissions (as expected) was Machine Translation. We promoted Social Media as a new area.

Twenty nine Area Chairs worked with 665 reviewers, producing 1830 long paper reviews and 1187 short paper reviews. Everything ran to a tight schedule and there were no slippages. This would not have been possible without our wonderful and diligent Area Chairs and Reviewers. Thanks!

We are delighted to have two keynote speakers, both of whom are very well known to the language community: Aravind Joshi and Mark Johnson. They will give coordinated talks addressing the 50th ACL anniversary: “Remembrance of ACLs past” and “Computational linguistics: Where do we go from here?” The ACL Lifetime Achievement Award will be announced on the last day of the conference.

Of the many papers, we selected two as being outstanding:

*Bayesian Symbol-Refined Tree Substitution Grammars for Syntactic Parsing*  
Hiroyuki Shindo, Yusuke Miyao, Akinori Fujino, Masaaki Nagata

*String Re-writing Kernel*  
Fan Bu, Hang Li, Xiaoyan Zhu

They will be presented as best papers in a dedicated session.

We thank the General Conference Chair Haizhou Li, the Local Arrangements Committee headed by Gary Geunbae Lee, Michael White and Maggie Li, the Publication Co-Chairs for coordinating and putting the proceedings together and all other committee chairs for their work. MO is especially thankful to Steve Clark for helpful tips on how to manage and run the whole process.

We hope you enjoy the conference!

Chin-Yew Lin, Microsoft Research Asia  
Miles Osborne, University of Edinburgh



# Organizing Committee

## General Chair

Haizhou Li, Institute for Infocomm Research

## Program Co-Chairs

Chin-Yew Lin, Microsoft Research Asia

Miles Osborne, University of Edinburgh

## Local Arrangement Co-Chairs

Gary Geunbae Lee, Pohang University of Science and Technology (POSTECH)

Jong C. Park, Korea Advanced Institute of Science and Technology (KAIST)

## Workshop Co-Chairs

Massimo Poesio, University of Essex

Satoshi Sekine, New York University

## Publication Co-Chairs

Maggie Li, The Hong Kong Polytechnic University

Michael White, The Ohio State University

## Publicity Chairs

Jung-jae Kim, Nanyang Technological University

Youngjoong Ko, Dong-A University

## Tutorial Chair

Michael Strube, HITS gmbH

## Demo Chair

Min Zhang, Institute for Infocomm Research

## Special Session Chair

Rafael E. Banchs, Institute for Infocomm Research

## Mentoring Service Chair

Joyce Chai, Michigan State University

## Exhibit Chair

Byeongchang Kim, Catholic University of Daegu

## Faculty Advisors (Student Research Workshop)

Kentaro Inui, Tohoku University

Greg Kondrak, University of Alberta

Yang Liu, University of Texas at Dallas

### **Student Chairs (Student Research Workshop)**

Jackie Cheung, University of Toronto  
Jun Hatori, University of Tokyo  
Carlos Henriquez, Technical University of Catalonia (UPC)  
Ann Irvine, Johns Hopkins University

### **Sponsorship Chairs**

Eiichiro Sumita, NICT  
Haifeng Wang, Baidu  
Michael Gamon, Microsoft  
Patrick Pantel, Microsoft  
Massimiliano Ciaramita, Google  
Idan Szpektor, Yahoo!

### **Local Arrangements Committee**

Gary Geunbae Lee (co-chair)  
Jong C. Park (co-chair)  
Jeong-Won Cha (social activities)  
Hanmin Jung (local sponsorship)  
Seungshik Kang (local finance)  
Byeongchang Kim (local exhibit)  
Harksoo Kim (government sponsorship)  
Jung-jae Kim (conference handbook)  
Youngjoong Ko (web site and flyer)  
Heuseok Lim (student volunteer management)  
Seong-Bae Park (internet, wifi and equipments)

### **Business Manager**

Priscilla Rasmussen



# Program Committee

## Program Co-chairs

Chin-Yew Lin, Microsoft Research Asia  
Miles Osborne, University of Edinburgh

## Area Chairs

Hongyuan Zha, School of Computational Science and Engineering, College of Computing, Georgia Institute of Technology  
Hsin-Hsi Chen, Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan  
Kemal Oflazer, Carnegie Mellon University - Qatar  
Mikio Nakano, Honda Research Institute, Japan  
Andrei Popescu-Belis, Idiap Research Institute, Switzerland  
Eneko Agirre, University of the Basque Country, Spain  
Jason Baldridge, The University of Texas at Austin, USA  
David Weir, University of Sussex, UK  
Trevor Cohn, University of Sheffield, UK  
Mark Dredze, Johns Hopkins University, USA  
Noah Smith, Carnegie Mellon University, USA  
Jan Wiebe, University of Pittsburgh, USA  
Daniel M. Bikel, Google Research, USA  
Mona T. Diab, Center for Computational Learning Systems, Columbia University, USA  
Fei Xia, Univ. of Washington, Seattle, USA  
Marcello Federico, Fondazione Bruno Kessler, Trento, Italy  
Adam Lopez, Human Language Technology Center of Excellence, Johns Hopkins University, USA  
David Talbot, Google Research, USA  
Hua Wu, Baidu, China  
Evgeniy Gabrilovich, Google, USA  
Naoaki Okazaki, Graduate School of Information Sciences, Tohoku University, Japan  
Stephen Clark, University of Cambridge, UK  
Liang Huang, University Southern California, USA  
Anja Belz, School of Computing, Engineering and Maths, University of Brighton, UK  
Ani Nenkova, University of Pennsylvania, USA  
Chung-Hsien Wu, Department of Computer Science and Information Engineering, National Cheng Kung University, TAIWAN  
Jian Su, Institute for Infocomm Research, Singapore  
Jianfeng Gao, Microsoft Research, Redmond, USA  
Vincent Ng, University of Texas at Dallas, USA

## Program Committee

Apoorv Agarwal, Lars Ahrenberg, Hua Ai, Cem Akkaya, Inaki Alegria, Jan Alexandersson, Enrique Alfonseca, Ben Allison, Sophia Ananiadou, Abhishek Arun, Giuseppe Attardi, Necip Fazil Ayan, Cecilia Ovesdotter Alm

Jing Bai, Collin Baker, Kirk Baker, Jason Baldrige, Breck Baldwin, Tim Baldwin, Carmen Banea, Marco Baroni, Regina Barzilay, Roberto Basili, Tilman Becker, Michael Bendersky, Taylor Berg-Kirkpatrick, Sabine Bergler, Shane Bergsma, Indrajit Bhattacharya, Pushpak Bhat-tacharyya, Archana Bhattarai, Jiang Bian, Gann Bierner, Ann Bies, Arianna Bisazza, John Blitzer, Michael Bloodgood, Phil Blunsom, Bernd Bohnet, Ondrej Bojar, Danushka Bollegala, Fran-cis Bond, Kalina Bontcheva, Stefano Borgo, Alexandre Bouchard, Jordan Boyd-Graber, Kristy Boyer, S.R.K. Branavan, Thorsten Brants, Chris Brew, Ted Briscoe, Samuel Brody, Sabine Buch-holz, Paul Buitelaar, Paula Buttery, William Byrne, Donna Byron, Olga babko-malaya, Antal van den Bosch

Aoife Cahill, Mary Elaine Califf, Chris Callison-Burch, Nicoletta Calzolari Zamorani, Nicola Cancedda, Yunbo Cao, Claire Cardie, Michael Carl, Xavier Carreras, John Carroll, Francisco Casacuberta, Vittorio Castelli, Asli Celikyilmaz, Mauro Cettolo, Joyce Chai, Nate Chambers, Yee Seng Chan, Ming-Wei Chang, Pi-Chuan Chang, Berlin Chen, Chia-Ping Chen, John Chen, Keh-Jiann Chen, Xueqi Cheng, Colin Cherry, David Chiang, Christian Chiercos, Hai Leong Chieu, Laura Chiticariu, Yejin Choi, Jennifer Chu-Carroll, Tat-Seng Chua, Ken Church, Alexan-der Clark, Shay Cohen, Trevor Cohn, Kevyn Collins-Thompson, Marta Ruiz Costa Jussa, Dan Cristea, Hang Cui, Aron Culotta, James Cussens, Chaoliu Chaoliu

Walter Daelemans, Ido Dagan, R. I. Damper, Cristian Danescu-Niculescu-Mizil, Van Dang, Lau-rence Danlos, Dipanjan Das, Hal Daume, Gerard De Melo, Guy De Pauw, Steve DeNeeffe, John DeNero, Vera Demberg, Yasuharu Den, Pascal Denis, Markus Dickinson, Mike Dillinger, Xi-aowen Ding, Kohji Dohsaka, John Dowding, Doug Downey, Markus Dreyer, Rebecca Dridan, Jinhua Du, Kevin Duh, Chris Dyer, Marc Dymetman

Judith Ecker-Kohler, Koji Eguchi, Andreas Eisele, Jacob Eisenstein, Jason Eisner, Michael El-hadad

Benoit Favre, Anna Feldman, Christiane Fellbaum, Raquel Fernandez, Margaret Fleck, Dan Flickinger, Radu Florian, Corina Forascu, Kate Forbes-Riley, George Foster, Anette Frank, Bob Frank, Dayne Freitag, Kotaro Funakoshi

Michel Galley, Michael Gamon, Kavita Ganesan, Juri Ganitkevitch, Wei Gao, Claire Gardent, Nikesh Garera, Michael Gasser, Albert Gatt, Dmitriy Genzel, Kallirroi Georgila, Sean Gerrish, Daniel Gildea, Jennifer Gillenwater, Dan Gillick, Kevin Gimpel, Roxana Girju, Claudio Giu-liano, Amir Globerson, Yoav Goldberg, Sharon Goldwater, Julio Gonzalo, Cyril Goutte, Joao Graca, Spence Green, Charlie Greenbacker, Stephan Greene, Ralph Grishman, Jiafeng Guo, Iryna Gurevych, Adria de Gispert, Josef van Genabith

Nizar Habash, Ben Hachey, Barry Haddow, Patrick Haffner, Dilek Hakkani-Tur, David Hall, Keith Hall, Xianpei Han, Jirka Hana, Sanda Harabagiu, Christian Hardmeier, Kazi Saidul Hasan, Sasa Hasan, Chikara Hashimoto, Koiti Hasida, Ahmed Hassan, Helen Hastie, Katsuhiko Hayashi, Xiaodong He, Zhongjun He, Jeffrey Heinz, John Henderson, Iris Hendrickx, Graeme Hirst, Hieu Hoang, Julia Hockenmaier, Matt Honnibal, Mark Hopkins, Veronique Hoste, Eduard Hovy, Paul Hsu, Fei Huang, Liang Huang, Minlie Huang, Ruihong Huang, Zhongqiang Huang, Mans Hulden

Nancy Ide, Gonzalo Iglesias, Ryu Iida, Nitin Indurkha, Grant Ingersoll, Diana Inkpen, Pierre Isabelle, Mitsuru Ishizuka, Tatsuya Izuha, Aranta Diaz de Ilarraza, Gary Lee

Jagadeesh Jagarlamudi, Heng Ji, Sittichai Jiampojarn, Jing Jiang, Wenbin Jiang, Richard Johansson, Howard Johnson, Mark Johnson, Rie Johnson, Doug Jones, Vanja Josifovski

Min-Yen Kan, Damianos Karakos, Daisuke Kawahara, Tatsuya Kawahara, Jun'ichi Kazama, Frank Keller, Andre Kempe, Shahram Khadivi, Emre Kiciman, Bernd Kiefer, Jungi Kim, Su Nam Kim, Katrin Kirchhoff, Ioannis Klapaftis, Thomas Kleinbauer, Alexandre Klementiev, Kevin Knight, Philipp Koehn, Rob Koeling, Oskar Kohonen, Kazunori Komatani, Greg Kondrak, Moshe Koppel, Anna Korhonen, Andras Kornai, Zornitsa Kozareva, Emiel Kraemer, Lun-Wei Ku, Sandra Kuebler, Marco Kuhlmann, Roland Kuhn, Seth Kulick, Tom Kwiatkowski, Oi Yee Kwong

Mikel L. Forcada, Wai Lam, Mathias Lambert, Felipe Langlais, Mirella Lapata, Alberto Lavelli, Alon Lavie, Yoong Keok Lee, Oliver Lemon, James Lester, Gregor Leusch, Gina-Anne Levow, Roger Levy, William Lewis, Fangtao Li, Haizhou Li, Hang Li, Shoushan Li, Xiao Li, Zhifei Li, Percy Liang, Shasha Liao, Chuan-Jie Lin, Ken Litkowski, Marina Litvak, Bing Liu, Fei Liu, Qun Liu, Yan Liu, Yang Liu, Yi Liu, Elena Lloret, Adam Lopez, Annie Louis, Xiaofei Lu, Yue Lu, Yajuan Lv, Keith vander Linden

Bin Ma, Yanjun Ma, Klaus Macherey, Wolfgang Macherey, Nitin Madnani, Suresh Manandhar, Gideon Mann, Christopher Manning, Daniel Marcu, Katja Markert, Konstantin Markov, Erwin Marsi, Andre Martins, Yuval Marton, Spyros Matsoukas, Yuichiroh Matsubayashi, Yuji Matsumoto, Takuya Matsuzaki, Evgeny Matusov, Arne Mauser, Jon May, Diana Maynard, Andrew McCallum, Diana McCarthy, David McClosky, Kathy McCoy, Ryan McDonald, Tara McIntosh, Kathy McKeown, Paul McNamee, Arul Menezes, Paola Merlo, Donald Metzler, Adam Meyers, Haitao Mi, Jeff Mielke, Rada Mihalcea, Yusuke Miyao, Saif Mohammad, Emad Mohammed, Behrang Mohit, Karo Moilanen, Dan Moldovan, Christian Monson, Christof Monz, Taesun Moon, Robert Moore, Roser Morante, Hamish Morgan, Alessandro Moschitti, Smaranda Muresan, Gabriel Murray, Markos Mylonakis

Toshiaki Nakazawa, Preslav Nakov, Tahira Nasseem, Vivi Nastase, Roberto Navigli, Mark-Jan Nederhof, Graham Neubig, Günter Neumann, Hwee Tou Ng, Vincent Ng, Jian-Yun Nie, Zaiqing Nie, Joakim Nivre, Gertjan van Noord

Stephan Oepen, Jong-Hoon Oh, Manabu Okumura, Constantin Orasan

Martha Palmer, Sinno Pan, Bo Pang, Ivandre Paraboni, Christopher Parisien, Kristen P. Parton, Becky Passonneau, Siddharth Patwardhan, Michael Paul, Matthias Paulik, Adam Pauls, Adam Pease, Fuchun Peng, Jing Peng, Gerald Penn, Marco Pennacchiotti, Slav Petrov, Sasa Petrovic, Daniele Pighin, Manfred Pinkal, Elias Ponvert, Simone Paolo Ponzetto, Hoifung Poon, Andrei Popescu-Belis, Maja Popovic, Fred Popowich, Matt Post, Christopher Potts, David Powers, Sameer Pradhan, Rashmi Prasad, Daniel Preotiuc, Adam Przepiórkowski, Matthew Purver, James Pustejovsky, Sampo Pyysalo

Ariadna Quattoni

Rob Gaizauskas, Drago Radev, Dragomir Radev, Kira Radinsky, Hema Raghavan, Altaf Rahman, Daniel Ramage, Ganesh Ramakrishnan, Owen Rambow, Delip Rao, Ari Rappoport, Antoine Raux, Sujith Ravi, Emmanuel Rayner, Roi Reichart, Ehud Reiter, Norbert Reithinger, Sebastian Riedel, Jason Riesa, Stefan Riezler, German Rigau, Laura Rimell, Eric Ringger, Alan Ritter, Brian Roark, Horacio Rodríguez, Carolyn Rose, Andrew Rosenberg, Dan Roth, Vasile Rus, Alexander Rush, Graham Russell, Anton Rytting

Soo Ngee Koh, Kenji Sagae, Hassan Sajjad, Hasim Sak, Tapio Salakoski, Murat Saraclar, Anoop Sarkar, Sudeshna Sarkar, Christina Sauper, David Schlangen, Helmut Schmid, Nathan Schneider, William Schuler, Sabine Schulte im Walde, Hinrich Schütze, Satoshi Sekine, Jean Senellart, Violeta Seretan, Hendra Setiawan, Dipti Sharma, Libin Shen, Wade Shen, Shuming Shi, Eyal Shnarch, Luo Si, Candy Sidner, Michel Simard, Khalil Sima'an, Gabriel Skantze, Otakar Smrz, Rion Snow, Ben Snyder, Stephen Soderland, Yang Song, Youngin Song, Lucia Specia, Caroline Sporleder, Rohini Srihari, Manfred Stede, Mark Steedman, Josef Steinberger, Amanda Stent, Svetlana Stoyanchev, Veselin Stoyanov, Michael Strube, Keh-Yih Su, Fabian Suchanek, Weiwei Sun, Mihai Surdeanu, Hisami Suzuki, Jun Suzuki, Stan Szpakowicz, Idan Szpektor, Sien Moens, Diarmuid ó Séaghdha

Hiroya Takamura, Koichi Takeda, David Talbot, Partha Talukdar, Kumiko Tanaka-Ishii, Stefan Thater, Mariet Theune, Joerg Tiedemann, Christoph Tillmann, Ivan Titov, Cigdem Toprak, Kristina Toutanova, Roy Tromble, Junichi Tsujii, Yoshimasa Tsuruoka, Dan Tufis

Jakob Uszkoreit, Masao Utiyama

Benjamin Van Durme, Lucy Vanderwende, Vasudeva Varma, Tony Veale, Paola Velardi, Marc Vilain, David Vilar, Aline Villavicencio, Sami Virpioja, Andreas Vlachos, Piek Vossen

Marilyn Walker, Hanna Wallach, Michael Walsh, Stephen Wan, Xiaojun Wan, Haifeng Wang, Hsin-Min Wang, Leo Wanner, Taro Watanabe, Yotaro Watanabe, Bonnie Webber, Julie Weeds, Daniel S. Weld, Ben Wellner, Ji-Rong Wen, Michael Wiegand, Jason Williams, Theresa Wilson, Shuly Wintner, John Wong, Kam-Fai Wong, Tak-Lam Wong, Kristian Woodsend, Chung-Hsien Wu, Xianchao Wu, Fei Wu

Yunqing Xia, Lei Xie, Shasha Xie, Deyi Xiong, Gu Xu, Peng Xu, Nianwen Xue, Xiaobin Xue

Charles Yang, Muyun Yang, Shuang-Hong Yang, Roman Yangarber, Tae Yano, Alexander Yates, Xing Yi, Scott Wen-Tau Yih, Anssi Yli-Jyra, Dong Yu, Kai Yu, Liang-Chih Yu, Yisong Yue, Deniz Yuret, Yichang Yichang

Fabio Zanzotto, Jakub Zavrel, Klaus Zechner, Dmitry Zelenko, Richard Zens, Torsten Zesch, Luke Zettlemoyer, ChengXiang Zhai, Bing Zhang, Duo Zhang, Hui Zhang, Joy Zhang, Min Zhang, Qi Zhang, Yi Zhang, Yue Zhang, Liu Zhanyi, Bing Zhao, Jun Zhao, Shiqi Zhao, Tiejun Zhao, Jing Zheng, Guodong Zhou, Qiang Zhou, Michael Zock, Ingrid Zukerman

## **Invited Talk**

### **Remembrance of ACLs past**

**Aravind K. Joshi**

Henry Salvatori Professor of Computer and Cognitive Science  
University of Pennsylvania

#### **Abstract**

Besides briefly covering some highlights of the past 50 years of ACL from my perspective, I will try to comment on (1) why some directions of research were pursued for a while and then dropped, sometimes for a good reason and sometimes apparently for no reason, (2) why the relationship to Linguistics, Psycholinguistics, and AI goes up and down, and (3) are there any leftovers that have the possibility of being turned into delicious contributions!

#### **Short Bio**

After completing his undergraduate work in Electrical and Communication Engineering in India, Aravind Joshi came to the University of Pennsylvania and obtained his Ph.D. in Electrical Engineering in 1960. At present, he is the Henry Salvatori Professor of Computer and Cognitive Science at the University of Pennsylvania.

Joshi has worked on several problems that overlap computer science and linguistics. More specifically, he has worked on topics in mathematical linguistics as they relate to formal and linguistic adequacy of different formalisms and their processing implications. He has also worked on several aspects of theories of representation and inference in natural language, especially as they relate to discourse.

Joshi was the President of ACL in 1975 and was appointed as a Founding Fellow of ACL in 2011. He was awarded the Lifetime Achievement Award of ACL in 2002, the David Rumelhart Prize of the Cognitive Science Society in 2003 and the Franklin Medal for Computer and Cognitive Science, Franklin Institute, Philadelphia, in 2005.

## **Invited Talk**

### **Computational linguistics: Where do we go from here?**

**Mark Johnson**

Professor of Language Sciences (CORE)  
Director, Centre for Language Sciences (CLaS)  
Department of Computing Faculty of Science  
Macquarie University  
Sydney, Australia

“Prediction is very difficult, especially about the future” —Niels Bohr

### **Abstract**

The very fact that we’re having a 50th annual meeting means that our field hasn’t been a complete failure, but will there still be computational linguistics meetings in 50 years time? How do we fit into the larger intellectual picture, and what would it take to make computational linguistics into a real engineering discipline, or, for that matter, a scientific one? Prognosticating fearlessly (or perhaps just foolishly) I’ll draw some lessons from the last 50 years about what the next few might hold.

### **Short Bio**

Mark Johnson is a Professor of Language Science (CORE) in the Department of Computing at Macquarie University. He was awarded a BSc (Hons) in 1979 from the University of Sydney, an MA in 1984 from the University of California, San Diego and a PhD in 1987 from Stanford University. He held a postdoctoral fellowship at MIT from 1987 until 1988, and has been a visiting researcher at the University of Stuttgart, the Xerox Research Centre in Grenoble, CSAIL at MIT and the Natural Language group at Microsoft Research. He has worked on a wide range of topics in computational linguistics, but his main research area is parsing and its applications to text and speech processing. He was President of the Association for Computational Linguistics in 2003, and was a professor from 1989 until 2009 in the Departments of Cognitive and Linguistic Sciences and Computer Science at Brown University.

## Table of Contents

<i>Learning to Translate with Multiple Objectives</i>	
Kevin Duh, Katsuhito Sudoh, Xianchao Wu, Hajime Tsukada and Masaaki Nagata . . . . .	1
<i>Joint Feature Selection in Distributed Stochastic Learning for Large-Scale Discriminative Training in SMT</i>	
Patrick Simianer, Stefan Riezler and Chris Dyer . . . . .	11
<i>Prediction of Learning Curves in Machine Translation</i>	
Prasanth Kolachina, Nicola Cancedda, Marc Dymetman and Sriram Venkatapathy . . . . .	22
<i>Probabilistic Integration of Partial Lexical Information for Noise Robust Haptic Voice Recognition</i>	
Khe Chai Sim . . . . .	31
<i>A Nonparametric Bayesian Approach to Acoustic Model Discovery</i>	
Chia-ying Lee and James Glass . . . . .	40
<i>Automated Essay Scoring Based on Finite State Transducer: towards ASR Transcription of Oral English Speech</i>	
Xingyuan Peng, Dengfeng Ke and Bo Xu . . . . .	50
<i>Text-level Discourse Parsing with Rich Linguistic Features</i>	
Vanessa Wei Feng and Graeme Hirst . . . . .	60
<i>PDTB-style Discourse Annotation of Chinese Text</i>	
Yuping Zhou and Nianwen Xue . . . . .	69
<i>SITS: A Hierarchical Nonparametric Model using Speaker Identity for Topic Segmentation in Multi-party Conversations</i>	
Viet-An Nguyen, Jordan Boyd-Graber and Philip Resnik . . . . .	78
<i>Extracting Narrative Timelines as Temporal Dependency Structures</i>	
Oleksandr Kolomiyets, Steven Bethard and Marie-Francine Moens . . . . .	88
<i>Labeling Documents with Timestamps: Learning from their Time Expressions</i>	
Nathanael Chambers . . . . .	98
<i>Temporally Anchored Relation Extraction</i>	
Guillermo Garrido, Anselmo Peñas, Bernardo Cabaleiro and Álvaro Rodrigo . . . . .	107
<i>Efficient Tree-based Approximation for Entailment Graph Learning</i>	
Jonathan Berant, Ido Dagan, Meni Adler and Jacob Goldberger . . . . .	117
<i>Learning High-Level Planning from Text</i>	
S.R.K. Branavan, Nate Kushman, Tao Lei and Regina Barzilay . . . . .	126

<i>Distributional Semantics in Technicolor</i>	
Elia Bruni, Gemma Boleda, Marco Baroni and Nam Khanh Tran . . . . .	136
<i>A Class-Based Agreement Model for Generating Accurately Inflected Translations</i>	
Spence Green and John DeNero . . . . .	146
<i>Deciphering Foreign Language by Combining Language Models and Context Vectors</i>	
Malte Nuhn, Arne Mauser and Hermann Ney . . . . .	156
<i>Machine Translation without Words through Substring Alignment</i>	
Graham Neubig, Taro Watanabe, Shinsuke Mori and Tatsuya Kawahara . . . . .	165
<i>Fast Syntactic Analysis for Statistical Language Modeling via Substructure Sharing and Uptraining</i>	
Ariya Rastrow, Mark Dredze and Sanjeev Khudanpur . . . . .	175
<i>Bootstrapping a Unified Model of Lexical and Phonetic Acquisition</i>	
Micha Elsner, Sharon Goldwater and Jacob Eisenstein . . . . .	184
<i>Discriminative Pronunciation Modeling: A Large-Margin, Feature-Rich Approach</i>	
Hao Tang, Joseph Keshet and Karen Livescu . . . . .	194
<i>Discriminative Strategies to Integrate Multiword Expression Recognition and Parsing</i>	
Matthieu Constant, Anthony Sigogne and Patrick Watrin . . . . .	204
<i>Utilizing Dependency Language Models for Graph-based Dependency Parsing Models</i>	
Wenliang Chen, Min Zhang and Haizhou Li . . . . .	213
<i>Spectral Learning of Latent-Variable PCFGs</i>	
Shay B. Cohen, Karl Stratos, Michael Collins, Dean P. Foster and Lyle Ungar . . . . .	223
<i>Reducing Approximation and Estimation Errors for Chinese Lexical Processing with Heterogeneous Annotations</i>	
Weiwei Sun and Xiaojun Wan . . . . .	232
<i>Capturing Paradigmatic and Syntagmatic Lexical Relations: Towards Accurate Chinese Part-of-Speech Tagging</i>	
Weiwei Sun and Hans Uszkoreit . . . . .	242
<i>Fast Online Training with Frequency-Adaptive Learning Rates for Chinese Word Segmentation and New Word Detection</i>	
Xu Sun, Houfeng Wang and Wenjie Li . . . . .	253
<i>Verb Classification using Distributional Similarity in Syntactic and Semantic Structures</i>	
Danilo Croce, Alessandro Moschitti, Roberto Basili and Martha Palmer . . . . .	263
<i>Word Sense Disambiguation Improves Information Retrieval</i>	
Zhi Zhong and Hwee Tou Ng . . . . .	273
<i>Efficient Search for Transformation-based Inference</i>	
Asher Stern, Roni Stern, Ido Dagan and Ariel Felner . . . . .	283



<i>Maximum Expected BLEU Training of Phrase and Lexicon Translation Models</i> Xiaodong He and Li Deng .....	292
<i>Learning Translation Consensus with Structured Label Propagation</i> Shujie Liu, Chi-Ho Li, Mu Li and Ming Zhou .....	302
<i>Smaller Alignment Models for Better Translations: Unsupervised Word Alignment with the l0-norm</i> Ashish Vaswani, Liang Huang and David Chiang .....	311
<i>Modeling Review Comments</i> Arjun Mukherjee and Bing Liu .....	320
<i>A Joint Model for Discovery of Aspects in Utterances</i> Asli Celikyilmaz and Dilek Hakkani-Tur .....	330
<i>Aspect Extraction through Semi-Supervised Modeling</i> Arjun Mukherjee and Bing Liu .....	339
<i>Learning to "Read Between the Lines" using Bayesian Logic Programs</i> Sindhu Raghavan, Raymond Mooney and Hyeonseoo Ku .....	349
<i>Collective Generation of Natural Image Descriptions</i> Polina Kuznetsova, Vicente Ordonez, Alexander Berg, Tamara Berg and Yejin Choi .....	359
<i>Concept-to-text Generation via Discriminative Reranking</i> Ioannis Konstas and Mirella Lapata .....	369
<i>A Discriminative Hierarchical Model for Fast Coreference at Large Scale</i> Michael Wick, Sameer Singh and Andrew McCallum .....	379
<i>Coreference Semantics from Web Features</i> Mohit Bansal and Dan Klein .....	389
<i>Subgroup Detection in Ideological Discussions</i> Amjad Abu-Jbara, Pradeep Dasigi, Mona Diab and Dragomir Radev .....	399
<i>Cross-Domain Co-Extraction of Sentiment and Topic Lexicons</i> Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang and Xiaoyan Zhu .....	410
<i>Learning Syntactic Verb Frames using Graphical Models</i> Thomas Lippincott, Anna Korhonen and Diarmuid Ó Séaghdha .....	420
<i>Fast Online Lexicon Learning for Grounded Language Acquisition</i> David Chen .....	430
<i>Bayesian Symbol-Refined Tree Substitution Grammars for Syntactic Parsing</i> Hiroyuki Shindo, Yusuke Miyao, Akinori Fujino and Masaaki Nagata .....	440
<i>String Re-writing Kernel</i> Fan Bu, Hang Li and Xiaoyan Zhu .....	449

<i>Translation Model Adaptation for Statistical Machine Translation with Monolingual Topic Information</i> Jinsong Su, Hua Wu, Haifeng Wang, Yidong Chen, Xiaodong Shi, Huailin Dong and Qun Liu	459
<i>A Statistical Model for Unsupervised and Semi-supervised Transliteration Mining</i> Hassan Sajjad, Alexander Fraser and Helmut Schmid	469
<i>Modified Distortion Matrices for Phrase-Based Statistical Machine Translation</i> Arianna Bisazza and Marcello Federico	478
<i>Semantic Parsing with Bayesian Tree Transducers</i> Bevan Jones, Mark Johnson and Sharon Goldwater	488
<i>Dependency Hashing for n-best CCG Parsing</i> Dominick Ng and James R. Curran	497
<i>Strong Lexicalization of Tree Adjoining Grammars</i> Andreas Maletti and Joost Engelfriet	506
<i>Tweet Recommendation with Graph Co-Ranking</i> Rui Yan, Mirella Lapata and Xiaoming Li	516
<i>Joint Inference of Named Entity Recognition and Normalization for Tweets</i> Xiaohua Liu, Ming Zhou, Xiangyang Zhou, Zhongyang Fu and Furu Wei	526
<i>Finding Bursty Topics from Microblogs</i> Qiming Diao, Jing Jiang, Feida Zhu and Ee-Peng Lim	536
<i>Spice it up? Mining Refinements to Online Instructions from User Generated Content</i> Gregory Druck and Bo Pang	545
<i>Sentence Dependency Tagging in Online Question Answering Forums</i> Zhonghua Qu and Yang Liu	554
<i>Mining Entity Types from Query Logs via User Intent Modeling</i> Patrick Pantel, Thomas Lin and Michael Gamon	563
<i>Cross-Lingual Mixture Model for Sentiment Classification</i> Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Ge Xu and Houfeng Wang	572
<i>Community Answer Summarization for Multi-Sentence Question with Group L1 Regularization</i> Wen Chan, Xiangdong Zhou, Wei Wang and Tat-Seng Chua	582
<i>Error Mining on Dependency Trees</i> Claire Gardent and Shashi Narayan	592
<i>Computational Approaches to Sentence Completion</i> Geoffrey Zweig, John C. Platt, Christopher Meek, Christopher J.C. Burges, Ainur Yessenalina and Qiang Liu	601

<i>Iterative Viterbi A* Algorithm for K-Best Sequential Decoding</i>	
Zhiheng Huang, Yi Chang, Bo Long, Jean-Francois Crespo, Anlei Dong, Sathiya Keerthi and Su-Lin Wu .....	611
<i>Bootstrapping via Graph Propagation</i>	
Max Whitney and Anoop Sarkar .....	620
<i>Selective Sharing for Multilingual Dependency Parsing</i>	
Tahira Naseem, Regina Barzilay and Amir Globerson .....	629
<i>The Creation of a Corpus of English Metalanguage</i>	
Shomir Wilson .....	638
<i>Crosslingual Induction of Semantic Roles</i>	
Ivan Titov and Alexandre Klementiev .....	647
<i>Head-driven Transition-based Parsing with Top-down Prediction</i>	
Katsuhiko Hayashi, Taro Watanabe, Masayuki Asahara and Yuji Matsumoto .....	657
<i>MIX Is Not a Tree-Adjoining Language</i>	
Makoto Kanazawa and Sylvain Salvati .....	666
<i>Exploiting Multiple Treebanks for Parsing with Quasi-synchronous Grammars</i>	
Zhenghua Li, Ting Liu and Wanxiang Che .....	675
<i>A Probabilistic Model for Canonicalizing Named Entity Mentions</i>	
Dani Yogatama, Yanchuan Sim and Noah A. Smith .....	685
<i>Multilingual Named Entity Recognition using Parallel Data and Metadata from Wikipedia</i>	
Sungchul Kim, Kristina Toutanova and Hwanjo Yu .....	694
<i>A Computational Approach to the Automation of Creative Naming</i>	
Gozde Ozbal and Carlo Strapparava .....	703
<i>Unsupervised Relation Discovery with Sense Disambiguation</i>	
Limin Yao, Sebastian Riedel and Andrew McCallum .....	712
<i>Reducing Wrong Labels in Distant Supervision for Relation Extraction</i>	
Shingo Takamatsu, Issei Sato and Hiroshi Nakagawa .....	721
<i>Finding Salient Dates for Building Thematic Timelines</i>	
Rémy Kessler, Xavier Tannier, Caroline Hagège, Véronique Moriceau and André Bittar .....	730
<i>Historical Analysis of Legal Opinions with a Sparse Mixed-Effects Latent Variable Model</i>	
William Yang Wang, Elijah Mayfield, Suresh Naidu and Jeremiah Dittmar .....	740
<i>A Topic Similarity Model for Hierarchical Phrase-based Translation</i>	
Xinyan Xiao, Deyi Xiong, Min Zhang, Qun Liu and Shouxun Lin .....	750

<i>Modeling Topic Dependencies in Hierarchical Text Categorization</i>	
Alessandro Moschitti, Qi Ju and Richard Johansson .....	759
<i>Attacking Parsing Bottlenecks with Unlabeled Data and Relevant Factorizations</i>	
Emily Pitler .....	768
<i>Semi-supervised Dependency Parsing using Lexical Affinities</i>	
Seyed Abolghasem Mirroshandel, Alexis Nasr and Joseph Le Roux .....	777
<i>Chinese Comma Disambiguation for Discourse Analysis</i>	
Yaqin Yang and Nianwen Xue .....	786
<i>Collective Classification for Fine-grained Information Status</i>	
Katja Markert, Yufang Hou and Michael Strube .....	795
<i>Structuring E-Commerce Inventory</i>	
Karin Mauge, Khash Rohanimanesh and Jean-David Ruvini .....	805
<i>Named Entity Disambiguation in Streaming Data</i>	
Alexandre Davis, Adriano Veloso, Altigran Soares, Alberto Laender and Wagner Meira Jr. ....	815
<i>Big Data versus the Crowd: Looking for Relationships in All the Right Places</i>	
Ce Zhang, Feng Niu, Christopher Ré and Jude Shavlik .....	825
<i>Automatic Event Extraction with Structured Preference Modeling</i>	
Wei Lu and Dan Roth .....	835
<i>Discriminative Learning for Joint Template Filling</i>	
Einat Minkov and Luke Zettlemoyer .....	845
<i>Classifying French Verbs Using French and English Lexical Resources</i>	
Ingrid Falk, Claire Gardent and Jean-Charles Lamirel .....	854
<i>Modeling Sentences in the Latent Space</i>	
Weimei Guo and Mona Diab .....	864
<i>Improving Word Representations via Global Context and Multiple Word Prototypes</i>	
Eric Huang, Richard Socher, Christopher Manning and Andrew Ng .....	873
<i>Exploiting Social Information in Grounded Language Learning via Grammatical Reduction</i>	
Mark Johnson, Katherine Demuth and Michael Frank .....	883
<i>You Had Me at Hello: How Phrasing Affects Memorability</i>	
Cristian Danescu-Niculescu-Mizil, Justin Cheng, Jon Kleinberg and Lillian Lee .....	892
<i>Modeling the Translation of Predicate-Argument Structure for SMT</i>	
Deyi Xiong, Min Zhang and Haizhou Li .....	902
<i>A Ranking-based Approach to Word Reordering for Statistical Machine Translation</i>	
Nan Yang, Mu Li, Dongdong Zhang and Nenghai Yu .....	912

<i>Character-Level Machine Translation Evaluation for Languages with Ambiguous Word Boundaries</i> Chang Liu and Hwee Tou Ng .....	921
<i>PORT: a Precision-Order-Recall MT Evaluation Metric for Tuning</i> Boxing Chen, Roland Kuhn and Samuel Larkin .....	930
<i>Mixing Multiple Translation Models in Statistical Machine Translation</i> Majid Razmara, George Foster, Baskaran Sankaran and Anoop Sarkar .....	940
<i>Hierarchical Chunk-to-String Translation</i> Yang Feng, Dongdong Zhang, Mu Li and Qun Liu .....	950
<i>Large-Scale Syntactic Language Modeling with Treelets</i> Adam Pauls and Dan Klein .....	959
<i>Text Segmentation by Language Using Minimum Description Length</i> Hiroshi Yamaguchi and Kumiko Tanaka-Ishii .....	969
<i>Improve SMT Quality with Automatically Extracted Paraphrase Rules</i> Wei He, Hua Wu, Haifeng Wang and Ting Liu .....	979
<i>Ecological Evaluation of Persuasive Messages Using Google AdWords</i> Marco Guerini, Carlo Strapparava and Oliviero Stock .....	988
<i>Polarity Consistency Checking for Sentiment Dictionaries</i> Eduard Dragut, Hong Wang, Clement Yu, Prasad Sistla and Weiyi Meng .....	997
<i>Combining Coherence Models and Machine Translation Evaluation Metrics for Summarization Evaluation</i> Ziheng Lin, Chang Liu, Hwee Tou Ng and Min-Yen Kan .....	1006
<i>Sentence Simplification by Monolingual Machine Translation</i> Sander Wubben, Antal van den Bosch and Emiel Krahmer .....	1015
<i>A Cost Sensitive Part-of-Speech Tagging: Differentiating Serious Errors from Minor Errors</i> Hyun-Je Song, Jeong-Woo Son, Tae-Gil Noh, Seong-Bae Park and Sang-Jo Lee .....	1025
<i>A Broad-Coverage Normalization System for Social Media Language</i> Fei Liu, Fuliang Weng and Xiao Jiang .....	1035
<i>Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese</i> Jun Hatori, Takuya Matsuzaki, Yusuke Miyao and Jun'ichi Tsujii .....	1045
<i>Exploring Deterministic Constraints: from a Constrained English POS Tagger to an Efficient ILP Solution to Chinese Word Segmentation</i> Qiuye Zhao and Mitch Marcus .....	1054



# Conference Program

**Monday July 9, 2012**

**(9:00 – 10:30) Invited Talk: Remembrance of ACLs past, by Aravind K. Joshi**

**Session 1a: (11:00 – 12:30) Machine Translation**

*Learning to Translate with Multiple Objectives*

Kevin Duh, Katsuhito Sudoh, Xianchao Wu, Hajime Tsukada and Masaaki Nagata

*Joint Feature Selection in Distributed Stochastic Learning for Large-Scale Discriminative Training in SMT*

Patrick Simianer, Stefan Riezler and Chris Dyer

*Prediction of Learning Curves in Machine Translation*

Prasanth Kolachina, Nicola Cancedda, Marc Dymetman and Sriram Venkatapathy

**Session 1b: (11:00 – 12:30) Speech**

*Probabilistic Integration of Partial Lexical Information for Noise Robust Haptic Voice Recognition*

Khe Chai Sim

*A Nonparametric Bayesian Approach to Acoustic Model Discovery*

Chia-ying Lee and James Glass

*Automated Essay Scoring Based on Finite State Transducer: towards ASR Transcription of Oral English Speech*

Xingyuan Peng, Dengfeng Ke and Bo Xu

**Monday July 9, 2012 (continued)**

**Session 1c: (11:00 – 12:30) Discourse**

*Text-level Discourse Parsing with Rich Linguistic Features*

Vanessa Wei Feng and Graeme Hirst

*PDTB-style Discourse Annotation of Chinese Text*

Yuping Zhou and Nianwen Xue

*SITS: A Hierarchical Nonparametric Model using Speaker Identity for Topic Segmentation in Multiparty Conversations*

Viet-An Nguyen, Jordan Boyd-Graber and Philip Resnik

**Session 1d: (11:00 – 12:30) Time**

*Extracting Narrative Timelines as Temporal Dependency Structures*

Oleksandr Kolomiyets, Steven Bethard and Marie-Francine Moens

*Labeling Documents with Timestamps: Learning from their Time Expressions*

Nathanael Chambers

*Temporally Anchored Relation Extraction*

Guillermo Garrido, Anselmo Peñas, Bernardo Cabaleiro and Álvaro Rodrigo

**Session 1e: (11:00 – 12:30) Meaning**

*Efficient Tree-based Approximation for Entailment Graph Learning*

Jonathan Berant, Ido Dagan, Meni Adler and Jacob Goldberger

*Learning High-Level Planning from Text*

S.R.K. Branavan, Nate Kushman, Tao Lei and Regina Barzilay

*Distributional Semantics in Technicolor*

Elia Bruni, Gemma Boleda, Marco Baroni and Nam Khanh Tran



**Monday July 9, 2012 (continued)**

**Session 2a: (14:00 – 15:30) Machine Translation**

*A Class-Based Agreement Model for Generating Accurately Inflected Translations*

Spence Green and John DeNero

*Deciphering Foreign Language by Combining Language Models and Context Vectors*

Malte Nuhn, Arne Mauser and Hermann Ney

*Machine Translation without Words through Substring Alignment*

Graham Neubig, Taro Watanabe, Shinsuke Mori and Tatsuya Kawahara

**Session 2b: (14:00 – 15:30) Speech**

*Fast Syntactic Analysis for Statistical Language Modeling via Substructure Sharing and Uptraining*

Ariya Rastrow, Mark Dredze and Sanjeev Khudanpur

*Bootstrapping a Unified Model of Lexical and Phonetic Acquisition*

Micha Elsner, Sharon Goldwater and Jacob Eisenstein

*Discriminative Pronunciation Modeling: A Large-Margin, Feature-Rich Approach*

Hao Tang, Joseph Keshet and Karen Livescu

**Session 2c: (14:00 – 15:30) Parsing**

*Discriminative Strategies to Integrate Multiword Expression Recognition and Parsing*

Matthieu Constant, Anthony Sigogne and Patrick Watrin

*Utilizing Dependency Language Models for Graph-based Dependency Parsing Models*

Wenliang Chen, Min Zhang and Haizhou Li

*Spectral Learning of Latent-Variable PCFGs*

Shay B. Cohen, Karl Stratos, Michael Collins, Dean P. Foster and Lyle Ungar

**Monday July 9, 2012 (continued)**

**Session 2d: (14:00 – 15:30) Chinese lexical processing**

*Reducing Approximation and Estimation Errors for Chinese Lexical Processing with Heterogeneous Annotations*

Weiwei Sun and Xiaojun Wan

*Capturing Paradigmatic and Syntagmatic Lexical Relations: Towards Accurate Chinese Part-of-Speech Tagging*

Weiwei Sun and Hans Uszkoreit

*Fast Online Training with Frequency-Adaptive Learning Rates for Chinese Word Segmentation and New Word Detection*

Xu Sun, Houfeng Wang and Wenjie Li

**Session 2e: (14:00 – 15:30) Lexical semantics**

*Verb Classification using Distributional Similarity in Syntactic and Semantic Structures*

Danilo Croce, Alessandro Moschitti, Roberto Basili and Martha Palmer

*Word Sense Disambiguation Improves Information Retrieval*

Zhi Zhong and Hwee Tou Ng

*Efficient Search for Transformation-based Inference*

Asher Stern, Roni Stern, Ido Dagan and Ariel Felner

**Session 3a: (16:00 – 17:30) Machine Translation**

*Maximum Expected BLEU Training of Phrase and Lexicon Translation Models*

Xiaodong He and Li Deng

*Learning Translation Consensus with Structured Label Propagation*

Shujie Liu, Chi-Ho Li, Mu Li and Ming Zhou

*Smaller Alignment Models for Better Translations: Unsupervised Word Alignment with the  $l_0$ -norm*

Ashish Vaswani, Liang Huang and David Chiang

**Monday July 9, 2012 (continued)**

**Session 3b: (16:00 – 17:30) Aspect**

*Modeling Review Comments*

Arjun Mukherjee and Bing Liu

*A Joint Model for Discovery of Aspects in Utterances*

Asli Celikyilmaz and Dilek Hakkani-Tur

*Aspect Extraction through Semi-Supervised Modeling*

Arjun Mukherjee and Bing Liu

**Session 3c: (16:00 – 17:30) Generation and Machine Reading**

*Learning to "Read Between the Lines" using Bayesian Logic Programs*

Sindhu Raghavan, Raymond Mooney and Hyeonsoo Ku

*Collective Generation of Natural Image Descriptions*

Polina Kuznetsova, Vicente Ordonez, Alexander Berg, Tamara Berg and Yejin Choi

*Concept-to-text Generation via Discriminative Reranking*

Ioannis Konstas and Mirella Lapata

**Session 3d: (16:00 – 17:30) Dialogue and Discourse**

*A Discriminative Hierarchical Model for Fast Coreference at Large Scale*

Michael Wick, Sameer Singh and Andrew McCallum

*Coreference Semantics from Web Features*

Mohit Bansal and Dan Klein

*Subgroup Detection in Ideological Discussions*

Amjad Abu-Jbara, Pradeep Dasigi, Mona Diab and Dragomir Radev

**Monday July 9, 2012 (continued)**

**Session 3e: (16:00 – 17:30) Lexicon**

*Cross-Domain Co-Extraction of Sentiment and Topic Lexicons*

Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang and Xiaoyan Zhu

*Learning Syntactic Verb Frames using Graphical Models*

Thomas Lippincott, Anna Korhonen and Diarmuid Ó Séaghdha

*Fast Online Lexicon Learning for Grounded Language Acquisition*

David Chen

**(18:00 – 20:30) Poster Session**

**Tuesday July 10, 2012**

**(9:00 – 10:30) Best Paper Session**

*Bayesian Symbol-Refined Tree Substitution Grammars for Syntactic Parsing*

Hiroyuki Shindo, Yusuke Miyao, Akinori Fujino and Masaaki Nagata

*String Re-writing Kernel*

Fan Bu, Hang Li and Xiaoyan Zhu

**Session 4b: (11:00 – 12:30) Machine Translation**

*Translation Model Adaptation for Statistical Machine Translation with Monolingual Topic Information*

Jinsong Su, Hua Wu, Haifeng Wang, Yidong Chen, Xiaodong Shi, Huailin Dong and Qun Liu

*A Statistical Model for Unsupervised and Semi-supervised Transliteration Mining*

Hassan Sajjad, Alexander Fraser and Helmut Schmid

*Modified Distortion Matrices for Phrase-Based Statistical Machine Translation*

Arianna Bisazza and Marcello Federico

**Tuesday July 10, 2012 (continued)**

**Session 4c: (11:00 – 12:30) Parsing**

*Semantic Parsing with Bayesian Tree Transducers*  
Bevan Jones, Mark Johnson and Sharon Goldwater

*Dependency Hashing for n-best CCG Parsing*  
Dominick Ng and James R. Curran

*Strong Lexicalization of Tree Adjoining Grammars*  
Andreas Maletti and Joost Engelfriet

**Session 4d: (11:00 – 12:30) Social Media**

*Tweet Recommendation with Graph Co-Ranking*  
Rui Yan, Mirella Lapata and Xiaoming Li

*Joint Inference of Named Entity Recognition and Normalization for Tweets*  
Xiaohua Liu, Ming Zhou, Xiangyang Zhou, Zhongyang Fu and Furu Wei

*Finding Bursty Topics from Microblogs*  
Qiming Diao, Jing Jiang, Feida Zhu and Ee-Peng Lim

**Session 4e: (11:00 – 12:30) User generated content**

*Spice it up? Mining Refinements to Online Instructions from User Generated Content*  
Gregory Druck and Bo Pang

*Sentence Dependency Tagging in Online Question Answering Forums*  
Zhonghua Qu and Yang Liu

*Mining Entity Types from Query Logs via User Intent Modeling*  
Patrick Pantel, Thomas Lin and Michael Gamon

**Tuesday July 10, 2012 (continued)**

**Session 5b: (14:00 – 15:30) NLP Apps**

*Cross-Lingual Mixture Model for Sentiment Classification*

Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Ge Xu and Houfeng Wang

*Community Answer Summarization for Multi-Sentence Question with Group L1 Regularization*

Wen Chan, Xiangdong Zhou, Wei Wang and Tat-Seng Chua

*Error Mining on Dependency Trees*

Claire Gardent and Shashi Narayan

**Session 5c: (14:00 – 15:30) Machine Learning**

*Computational Approaches to Sentence Completion*

Geoffrey Zweig, John C. Platt, Christopher Meek, Christopher J.C. Burges, Ainur Yesse-  
nalina and Qiang Liu

*Iterative Viterbi A\* Algorithm for K-Best Sequential Decoding*

Zhiheng Huang, Yi Chang, Bo Long, Jean-Francois Crespo, Anlei Dong, Sathiya Keerthi  
and Su-Lin Wu

*Bootstrapping via Graph Propagation*

Max Whitney and Anoop Sarkar

**Session 5d: (14:00 – 15:30) Multilinguality**

*Selective Sharing for Multilingual Dependency Parsing*

Tahira Naseem, Regina Barzilay and Amir Globerson

*The Creation of a Corpus of English Metalanguage*

Shomir Wilson

*Crosslingual Induction of Semantic Roles*

Ivan Titov and Alexandre Klementiev

**Tuesday July 10, 2012 (continued)**

**Session 5e: (14:00 – 15:30) Parsing**

*Head-driven Transition-based Parsing with Top-down Prediction*

Katsuhiko Hayashi, Taro Watanabe, Masayuki Asahara and Yuji Matsumoto

*MIX Is Not a Tree-Adjoining Language*

Makoto Kanazawa and Sylvain Salvati

*Exploiting Multiple Treebanks for Parsing with Quasi-synchronous Grammars*

Zhenghua Li, Ting Liu and Wanxiang Che

**Session 6b: (16:00 – 17:30) Names**

*A Probabilistic Model for Canonicalizing Named Entity Mentions*

Dani Yogatama, Yanchuan Sim and Noah A. Smith

*Multilingual Named Entity Recognition using Parallel Data and Metadata from Wikipedia*

Sungchul Kim, Kristina Toutanova and Hwanjo Yu

*A Computational Approach to the Automation of Creative Naming*

Gozde Ozbek and Carlo Strapparava

**Session 6c: (16:00 – 17:30) Relations**

*Unsupervised Relation Discovery with Sense Disambiguation*

Limin Yao, Sebastian Riedel and Andrew McCallum

*Reducing Wrong Labels in Distant Supervision for Relation Extraction*

Shingo Takamatsu, Issei Sato and Hiroshi Nakagawa

*Finding Salient Dates for Building Thematic Timelines*

Rémy Kessler, Xavier Tannier, Caroline Hagège, Véronique Moriceau and André Bittar

**Tuesday July 10, 2012 (continued)**

**Session 6d: (16:00 – 17:30) Topics**

*Historical Analysis of Legal Opinions with a Sparse Mixed-Effects Latent Variable Model*

William Yang Wang, Elijah Mayfield, Suresh Naidu and Jeremiah Dittmar

*A Topic Similarity Model for Hierarchical Phrase-based Translation*

Xinyan Xiao, Deyi Xiong, Min Zhang, Qun Liu and Shouxun Lin

*Modeling Topic Dependencies in Hierarchical Text Categorization*

Alessandro Moschitti, Qi Ju and Richard Johansson

**Session 6e: (16:00 – 17:30) Parsing**

*Attacking Parsing Bottlenecks with Unlabeled Data and Relevant Factorizations*

Emily Pitler

*Semi-supervised Dependency Parsing using Lexical Affinities*

Seyed Abolghasem Mirroshandel, Alexis Nasr and Joseph Le Roux

**Wednesday July 11, 2012**

**(9:00 – 10:30) Invited Talk: Computational linguistics: Where do we go from here?,  
by Mark Johnson**

**Monday July 9, 2012**

**(18:00 – 20:30) Poster Session**

*Chinese Comma Disambiguation for Discourse Analysis*

Yaqin Yang and Nianwen Xue

*Collective Classification for Fine-grained Information Status*

Katja Markert, Yufang Hou and Michael Strube

*Structuring E-Commerce Inventory*

Karin Mauge, Khash Rohanimanesh and Jean-David Ruvini



**Monday July 9, 2012 (continued)**

*Named Entity Disambiguation in Streaming Data*

Alexandre Davis, Adriano Veloso, Altigran Soares, Alberto Laender and Wagner Meira Jr.

*Big Data versus the Crowd: Looking for Relationships in All the Right Places*

Ce Zhang, Feng Niu, Christopher Ré and Jude Shavlik

*Automatic Event Extraction with Structured Preference Modeling*

Wei Lu and Dan Roth

*Discriminative Learning for Joint Template Filling*

Einat Minkov and Luke Zettlemoyer

*Classifying French Verbs Using French and English Lexical Resources*

Ingrid Falk, Claire Gardent and Jean-Charles Lamirel

*Modeling Sentences in the Latent Space*

Weiwei Guo and Mona Diab

*Improving Word Representations via Global Context and Multiple Word Prototypes*

Eric Huang, Richard Socher, Christopher Manning and Andrew Ng

*Exploiting Social Information in Grounded Language Learning via Grammatical Reduction*

Mark Johnson, Katherine Demuth and Michael Frank

*You Had Me at Hello: How Phrasing Affects Memorability*

Cristian Danescu-Niculescu-Mizil, Justin Cheng, Jon Kleinberg and Lillian Lee

*Modeling the Translation of Predicate-Argument Structure for SMT*

Deyi Xiong, Min Zhang and Haizhou Li

*A Ranking-based Approach to Word Reordering for Statistical Machine Translation*

Nan Yang, Mu Li, Dongdong Zhang and Nenghai Yu

*Character-Level Machine Translation Evaluation for Languages with Ambiguous Word Boundaries*

Chang Liu and Hwee Tou Ng

**Monday July 9, 2012 (continued)**

*PORT: a Precision-Order-Recall MT Evaluation Metric for Tuning*

Boxing Chen, Roland Kuhn and Samuel Larkin

*Mixing Multiple Translation Models in Statistical Machine Translation*

Majid Razmara, George Foster, Baskaran Sankaran and Anoop Sarkar

*Hierarchical Chunk-to-String Translation*

Yang Feng, Dongdong Zhang, Mu Li and Qun Liu

*Large-Scale Syntactic Language Modeling with Treelets*

Adam Pauls and Dan Klein

*Text Segmentation by Language Using Minimum Description Length*

Hiroshi Yamaguchi and Kumiko Tanaka-Ishii

*Improve SMT Quality with Automatically Extracted Paraphrase Rules*

Wei He, Hua Wu, Haifeng Wang and Ting Liu

*Ecological Evaluation of Persuasive Messages Using Google AdWords*

Marco Guerini, Carlo Strapparava and Oliviero Stock

*Polarity Consistency Checking for Sentiment Dictionaries*

Eduard Dragut, Hong Wang, Clement Yu, Prasad Sistla and Weiyi Meng

*Combining Coherence Models and Machine Translation Evaluation Metrics for Summarization Evaluation*

Ziheng Lin, Chang Liu, Hwee Tou Ng and Min-Yen Kan

*Sentence Simplification by Monolingual Machine Translation*

Sander Wubben, Antal van den Bosch and Emiel Krahmer

*A Cost Sensitive Part-of-Speech Tagging: Differentiating Serious Errors from Minor Errors*

Hyun-Je Song, Jeong-Woo Son, Tae-Gil Noh, Seong-Bae Park and Sang-Jo Lee

*A Broad-Coverage Normalization System for Social Media Language*

Fei Liu, Fuliang Weng and Xiao Jiang

**Monday July 9, 2012 (continued)**

*Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese*

Jun Hatori, Takuya Matsuzaki, Yusuke Miyao and Jun'ichi Tsujii

*Exploring Deterministic Constraints: from a Constrained English POS Tagger to an Efficient ILP Solution to Chinese Word Segmentation*

Qiuye Zhao and Mitch Marcus



# Learning to Translate with Multiple Objectives

Kevin Duh\* Katsuhito Sudoh Xianchao Wu Hajime Tsukada Masaaki Nagata

NTT Communication Science Laboratories

2-4 Hikari-dai, Seika-cho, Kyoto 619-0237, JAPAN

kevinduh@is.naist.jp, lastname.firstname@lab.ntt.co.jp

## Abstract

We introduce an approach to optimize a machine translation (MT) system on multiple metrics *simultaneously*. Different metrics (e.g. BLEU, TER) focus on different aspects of translation quality; our multi-objective approach leverages these diverse aspects to improve overall quality.

Our approach is based on the theory of Pareto Optimality. It is simple to implement on top of existing single-objective optimization methods (e.g. MERT, PRO) and outperforms ad hoc alternatives based on linear-combination of metrics. We also discuss the issue of metric tunability and show that our Pareto approach is more effective in incorporating new metrics from MT evaluation for MT optimization.

## 1 Introduction

Weight optimization is an important step in building machine translation (MT) systems. Discriminative optimization methods such as MERT (Och, 2003), MIRA (Crammer et al., 2006), PRO (Hopkins and May, 2011), and Downhill-Simplex (Nelder and Mead, 1965) have been influential in improving MT systems in recent years. These methods are effective because they tune the system to maximize an automatic evaluation metric such as BLEU, which serve as surrogate objective for translation quality.

However, we know that a single metric such as BLEU is not enough. *Ideally*, we want to tune towards an automatic metric that has *perfect* correlation with human judgments of translation quality.

While many alternatives have been proposed, such a perfect evaluation metric remains elusive.

As a result, many MT evaluation campaigns now report multiple evaluation metrics (Callison-Burch et al., 2011; Paul, 2010). Different evaluation metrics focus on different aspects of translation quality. For example, while BLEU (Papineni et al., 2002) focuses on word-based n-gram precision, METEOR (Lavie and Agarwal, 2007) allows for stem/synonym matching and incorporates recall. TER (Snover et al., 2006) allows arbitrary chunk movements, while permutation metrics like RIBES (Isozaki et al., 2010; Birch et al., 2010) measure deviation in word order. Syntax (Owczarzak et al., 2007) and semantics (Pado et al., 2009) also help. Arguably, all these metrics correspond to our intuitions on what is a good translation.

The current approach of optimizing MT towards a single metric runs the risk of sacrificing other metrics. Can we really claim that a system is good if it has high BLEU, but very low METEOR? Similarly, is a high-METEOR low-BLEU system desirable? Our goal is to propose a multi-objective optimization method that avoids “overfitting to a single metric”. We want to build a MT system that does well with respect to many aspects of translation quality.

In general, we cannot expect to improve multiple metrics jointly if there are some inherent trade-offs. We therefore need to define the notion of Pareto Optimality (Pareto, 1906), which characterizes this tradeoff in a rigorous way and distinguishes the set of equally good solutions. We will describe Pareto Optimality in detail later, but roughly speaking, a

---

\*Now at Nara Institute of Science & Technology (NAIST)

hypothesis is pareto-optimal if there exist no other hypothesis better in all metrics. The contribution of this paper is two-fold:

- We introduce **PMO** (*Pareto-based Multi-objective Optimization*), a general approach for learning with multiple metrics. Existing single-objective methods can be easily extended to multi-objective using PMO.
- We show that PMO outperforms the alternative (single-objective optimization of linearly-combined metrics) in multi-objective space, and especially obtains stronger results for metrics that may be difficult to tune individually.

In the following, we first explain the theory of Pareto Optimality (Section 2), and then use it to build up our proposed PMO approach (Section 3). Experiments on NIST Chinese-English and PubMed English-Japanese translation using BLEU, TER, and RIBES are presented in Section 4. We conclude by discussing related work (Section 5) and opportunities/limitations (Section 6).

## 2 Theory of Pareto Optimality

### 2.1 Definitions and Concepts

The idea of Pareto optimality comes originally from economics (Pareto, 1906), where the goal is to characterize situations when a change in allocation of goods does not make anybody worse off. Here, we will explain it in terms of MT:

Let  $h \in L$  be a hypothesis from an N-best list  $L$ . We have a total of  $K$  different metrics  $M_k(h)$  for evaluating the quality of  $h$ . Without loss of generality, we assume metric scores are bounded between 0 and 1, with 1 being perfect. Each hypothesis  $h$  can be mapped to a  $K$ -dimensional vector  $M(h) = [M_1(h); M_2(h); \dots; M_K(h)]$ . For example, suppose  $K = 2$ ,  $M_1(h)$  computes the BLEU score, and  $M_2(h)$  gives the METEOR score of  $h$ . Figure 1 illustrates the set of vectors  $\{M(h)\}$  in a 10-best list.

For two hypotheses  $h_1, h_2$ , we write  $M(h_1) > M(h_2)$  if  $h_1$  is *better than*  $h_2$  in all metrics, and  $M(h_1) \geq M(h_2)$  if  $h_1$  is *better than or equal to*  $h_2$  in all metrics. When  $M(h_1) \geq M(h_2)$  and  $M_k(h_1) > M_k(h_2)$  for at least one metric  $k$ , we say that  $h_1$  *dominates*  $h_2$  and write  $M(h_1) \triangleright M(h_2)$ .

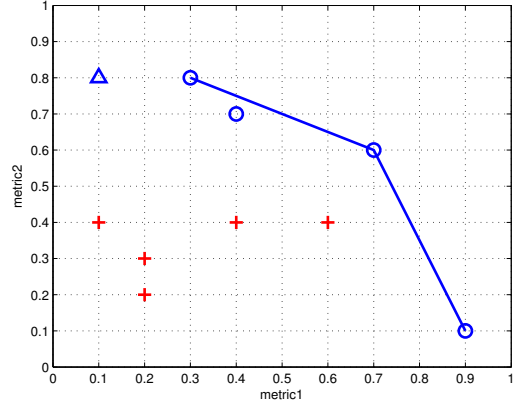


Figure 1: Illustration of Pareto Frontier. Ten hypotheses are plotted by their scores in two metrics. Hypotheses indicated by a circle (o) are pareto-optimal, while those indicated by a plus (+) are not. The line shows the convex hull, which attains only a subset of pareto-optimal points. The triangle ( $\Delta$ ) is a point that is weakly pareto-optimal but not pareto-optimal.

**Definition 1.** *Pareto Optimal:* A hypothesis  $h^* \in L$  is pareto-optimal iff there does not exist another hypothesis  $h \in L$  such that  $M(h) \triangleright M(h^*)$ .

In Figure 1, the hypotheses indicated by circle (o) are pareto-optimal, while those with plus (+) are not. To visualize this, take for instance the pareto-optimal point (0.4,0.7). There is no other point with either (metric1  $>$  0.4 and metric2  $\geq$  0.7), or (metric1  $\geq$  0.4 and metric2  $>$  0.7). On the other hand, the non-pareto point (0.6,0.4) is “dominated” by another point (0.7,0.6), because for metric1:  $0.7 > 0.6$  and for metric2:  $0.6 > 0.4$ .

There is another definition of optimality, which disregards ties and may be easier to visualize:

**Definition 2.** *Weakly Pareto Optimal:* A hypothesis  $h^* \in L$  is weakly pareto-optimal iff there is no other hypothesis  $h \in L$  such that  $M(h) > M(h^*)$ .

Weakly pareto-optimal points are a superset of pareto-optimal points. A hypothesis is weakly pareto-optimal if there is no other hypothesis that improves all the metrics; a hypothesis is pareto-optimal if there is no other hypothesis that improves at least one metric without detriment to other metrics. In Figure 1, point (0.1,0.8) is weakly pareto-optimal but not pareto-optimal, because of the competing point (0.3,0.8). Here we focus on pareto-optimality, but note our algorithms can be easily

modified for weakly pareto-optimality. Finally, we can introduce the key concept used in our proposed PMO approach:

**Definition 3. Pareto Frontier:** Given an  $N$ -best list  $L$ , the set of all pareto-optimal hypotheses  $h \in L$  is called the Pareto Frontier.

The Pareto Frontier has two desirable properties from the multi-objective optimization perspective:

1. Hypotheses on the Frontier are equivalently good in the Pareto sense.
2. For each hypothesis not on the Frontier, there is always a better (pareto-optimal) hypothesis.

This provides a principled approach to optimization: i.e. optimizing towards points on the Frontier and away from those that are not, and giving no preference to different pareto-optimal hypotheses.

## 2.2 Reduction to Linear Combination

Multi-objective problems can be formulated as:

$$\arg \max_w [M_1(h); M_2(h); \dots; M_k(h)] \quad (1)$$

where  $h = \text{Decode}(w, f)$

Here, the MT system's `Decode` function, parameterized by weight vector  $w$ , takes in a foreign sentence  $f$  and returns a translated hypothesis  $h$ . The `argmax` operates in vector space and our goal is to find  $w$  leading to hypotheses on the Pareto Frontier.

In the study of Pareto Optimality, one central question is: To what extent can multi-objective problems be solved by single-objective methods? Equation 1 can be *reduced* to a single-objective problem by scalarizing the vector  $[M_1(h); \dots; M_k(h)]$  with a linear combination:

$$\arg \max_w \sum_{k=1}^K p_k M_k(h) \quad (2)$$

where  $h = \text{Decode}(w, f)$

Here,  $p_k$  are positive real numbers indicating the relative importance of each metric (without loss of generality, assume  $\sum_k p_k = 1$ ). Are the solutions to Eq. 2 also solutions to Eq. 1 (i.e. pareto-optimal) and vice-versa? The theory says:

**Theorem 1. Sufficient Condition:** If  $w^*$  is solution to Eq. 2, then it is weakly pareto-optimal. Further, if  $w^*$  is unique, then it is pareto-optimal.

**Theorem 2. No Necessary Condition:** There may exist solutions to Eq. 1 that cannot be achieved by Eq. 2, irregardless of any setting of  $\{p_k\}$ .

Theorem 1 is a positive result asserting that linear combination can give pareto-optimal solutions. However, Theorem 2 states the limits: in particular, Eq. 2 attains only pareto-optimal points that are on the convex hull. This is illustrated in Figure 1: imagine sweeping all values of  $p_1 = [0, 1]$  and  $p_2 = 1 - p_1$  and recording the set of hypotheses that maximizes  $\sum_k p_k M_k(h)$ . For  $0.6 < p_1 \leq 1$  we get  $h = (0.9, 0.1)$ , for  $p_1 = 0.6$  we get  $(0.7, 0.6)$ , and for  $0 < p_1 < 0.6$  we get  $(0.4, 0.8)$ . At no setting of  $p_1$  do we attain  $h = (0.4, 0.7)$  which is also pareto-optimal but not on the convex hull.<sup>1</sup> This may have ramifications for issues like metric tunability and local optima. To summarize, linear combination is reasonable but has limitations. Our proposed approach will instead *directly* solve Eq. 1.

Pareto Optimality and multi-objective optimization is a deep field with active inquiry in engineering, operations research, economics, etc. For the interested reader, we recommend the survey by Marler and Arora (2004) and books by (Sawaragi et al., 1985; Miettinen, 1998).

## 3 Multi-objective Algorithms

### 3.1 Computing the Pareto Frontier

Our PMO approach will need to compute the Pareto Frontier for potentially large sets of points, so we first describe how this can be done efficiently. Given a set of  $N$  vectors  $\{M(h)\}$  from an  $N$ -best list  $L$ , our goal is extract the subset that are pareto-optimal.

Here we present an algorithm based on *iterative filtering*, in our opinion the simplest algorithm to understand and implement. The strategy is to loop through the list  $L$ , keeping track of any dominant points. Given a dominant point, it is easy to filter out many points that are dominated by it. After successive rounds, any remaining points that are not fil-

<sup>1</sup>We note that scalarization by *exponentiated*-combination  $\sum_k p_k M_k(h)^q$ , for a suitable  $q > 0$ , does satisfy necessary conditions for pareto optimality. However the proper tuning of  $q$  is not known a priori. See (Miettinen, 1998) for theorem proofs.

---

**Algorithm 1** FindParetoFrontier

---

**Input:**  $\{M(h)\}, h \in L$ **Output:** All pareto-optimal points of  $\{M(h)\}$ 

```
1:  $\mathcal{F} = \emptyset$ 
2: while  $L$  is not empty do
3:    $h^* = \text{shift}(L)$ 
4:   for each  $h$  in  $L$  do
5:     if  $(M(h^*) \triangleright M(h))$ : remove  $h$  from  $L$ 
6:     else if  $(M(h) \triangleright M(h^*))$ : remove  $h$  from  $L$ ; set  $h^* = h$ 
7:   end for
8:   Add  $h^*$  to Frontier Set  $\mathcal{F}$ 
9:   for each  $h$  in  $L$  do
10:    if  $(M(h^*) \triangleright M(h))$ : remove  $h$  from  $L$ 
11:  end for
12: end while
13: Return  $\mathcal{F}$ 
```

---

tered are necessarily pareto-optimal. Algorithm 1 shows the pseudocode. In line 3, we take a point  $h^*$  and check if it is dominating or dominated in the for-loop (lines 4-8). At least one pareto-optimal point will be found by line 8. The second loop (lines 9-11) further filters the list for points that are dominated by  $h^*$  but iterated before  $h^*$  in the first for-loop.

The outer while-loop stops exactly after  $P$  iterations, where  $P$  is the actual number of pareto-optimal points in  $L$ . Each inner loop costs  $O(KN)$  so the total complexity is  $O(PKN)$ . Since  $P \leq N$  with the actual value depending on the probability distribution of  $\{M(h)\}$ , the worst-case run-time is  $O(KN^2)$ . For a survey of various Pareto algorithms, refer to (Godfrey et al., 2007). The algorithm we described here is borrowed from the database literature in what is known as skyline operators.<sup>2</sup>

### 3.2 PMO-PRO Algorithm

We are now ready to present an algorithm for multi-objective optimization. As we will see, it can be seen as a generalization of the pairwise ranking optimization (PRO) of (Hopkins and May, 2011), so we call it PMO-PRO. PMO-PRO approach works by iteratively decoding-and-optimizing on the devset, sim-

---

<sup>2</sup>The inquisitive reader may wonder how is Pareto related to databases. The motivation is to incorporate preferences into relational queries (Börzsönyi et al., 2001). For  $K = 2$  metrics, they also present an alternative faster  $O(N \log N)$  algorithm by first topologically sorting along the 2 dimensions. All dominated points can be filtered by one-pass by comparing with the most-recent dominating point.

ilar to many MT optimization methods. The main difference is that rather than trying to maximize a single metric, we maximize the number of pareto points, in order to expand the Pareto Frontier

We will explain PMO-PRO in terms of the pseudo-code shown in Algorithm 2. For each sentence pair  $(f, e)$  in the devset, we first generate an N-best list  $L \equiv \{h\}$  using the current weight vector  $w$  (line 5). In line 6, we evaluate each hypothesis  $h$  with respect to the  $K$  metrics, giving a set of  $K$ -dimensional vectors  $\{M(h)\}$ .

Lines 7-8 is the critical part: it gives a “label” to each hypothesis, based on whether it is in the Pareto Frontier. In particular, first we call FindParetoFrontier (Algorithm 1), which returns a set of pareto hypotheses; pareto-optimal hypotheses will get label 1 while non-optimal hypotheses will get label 0. This information is added to the training set  $\mathcal{T}$  (line 8), which is then optimized by any conventional subroutine in line 10. We will follow PRO in using a pairwise classifier in line 10, which finds  $w^*$  that separates hypotheses with labels 1 vs. 0. In essence, this is the trick we employ to directly optimize on the Pareto Frontier. If we had used BLEU scores rather than the  $\{0, 1\}$  labels in line 8, the entire PMO-PRO algorithm would revert to single-objective PRO.

By definition, there is no single “best” result for multi-objective optimization, so we collect all weights and return the Pareto-optimal set. In line 13 we evaluate each weight  $w$  on  $K$  metrics across the entire corpus and call FindParetoFrontier in line 14.<sup>3</sup> This choice highlights an interesting change of philosophy: While setting  $\{p_k\}$  in linear-combination forces the designer to make an *a priori* preference among metrics prior to optimization, the PMO strategy is to optimize first agnostically and *a posteriori* let the designer choose among a set of weights. Arguably it is easier to choose among solutions based on their evaluation scores rather than devising exact values for  $\{p_k\}$ .

### 3.3 Discussion

**Variants:** In practice we find that a slight modification of line 8 in Algorithm 2 leads to more sta-

---

<sup>3</sup>Note this is the same FindParetoFrontier algorithm as used in line 7. Both operate on sets of *points* in  $K$ -dimensional space, induced from either weights  $\{w\}$  or hypotheses  $\{h\}$ .



---

**Algorithm 2** Proposed PMO-PRO algorithm

---

**Input:** Devset, max number of iterations  $I$ **Output:** A set of (pareto-optimal) weight vectors

```
1: Initialize  $w$ . Let  $\mathcal{W} = \emptyset$ .
2: for  $i = 1$  to  $I$  do
3:   Let  $\mathcal{T} = \emptyset$ .
4:   for each  $(f, e)$  in devset do
5:      $\{h\} = \text{DecodeNbest}(w, f)$ 
6:      $\{M(h)\} = \text{EvalMetricsOnSentence}(\{h\}, e)$ 
7:      $\{f\} = \text{FindParetoFrontier}(\{M(h)\})$ 
8:     foreach  $h \in \{h\}$ :
9:       if  $h \in \{f\}$ , set  $l=1$ , else  $l=0$ ; Add  $(l, h)$  to  $\mathcal{T}$ 
10:    end for
11:    $w^* = \text{OptimizationSubroutine}(\mathcal{T}, w)$ 
12:   Add  $w^*$  to  $\mathcal{W}$ ; Set  $w = w^*$ .
13: end for
14: Return  $\text{FindParetoFrontier}(\{M(w)\})$ 
```

---

ble results for PMO-PRO: for non-pareto hypotheses  $h \notin \{f\}$ , we set label  $l = \sum_k M_k(h)/K$  instead of  $l=0$ , so the method not only learns to discriminate pareto vs. non-pareto but also also learns to discriminate among competing non-pareto points. Also, like other MT works, in line 5 the N-best list is concatenated to N-best lists from previous iterations, so  $\{h\}$  is a set with  $i \cdot N$  elements.

**General PMO Approach:** The strategy we outlined in Section 3.2 can be easily applied to other MT optimization techniques. For example, by replacing the optimization subroutine (line 10, Algorithm 2) with a Powell search (Och, 2003), one can get PMO-MERT<sup>4</sup>. Alternatively, by using the large-margin optimizer in (Chiang et al., 2009) and moving it into the for-each loop (lines 4-9), one can get an online algorithm such PMO-MIRA. Virtually all MT optimization algorithms have a place where metric scores feedback into the optimization procedure; the idea of PMO is to replace these raw scores with labels derived from Pareto optimality.

## 4 Experiments

### 4.1 Evaluation Methodology

We experiment with two datasets: (1) The **PubMed** task is English-to-Japanese translation of scientific

<sup>4</sup>A difference with traditional MERT is the necessity of sentence-BLEU (Liang et al., 2006) in line 6. We use sentence-BLEU for optimization but corpus-BLEU for evaluation here.

abstracts. As metrics we use BLEU and RIBES (which demonstrated good human correlation in this language pair (Goto et al., 2011)). (2) The **NIST** task is Chinese-to-English translation with OpenMT08 training data and MT06 as devset. As metrics we use BLEU and NTER.

- BLEU =  $BP \times (\prod prec_n)^{1/4}$ . BP is brevity penalty.  $prec_n$  is precision of  $n$ -gram matches.
- RIBES =  $(\tau + 1)/2 \times prec_1^{1/4}$ , with Kendall’s  $\tau$  computed by measuring permutation between matching words in reference and hypothesis<sup>5</sup>.
- NTER =  $\max(1 - \text{TER}, 0)$ , which normalizes Translation Edit Rate<sup>6</sup> so that NTER=1 is best.

We compare two multi-objective approaches:

1. Linear-Combination of metrics (Eq. 2), optimized with PRO. We search a range of combination settings:  $(p_1, p_2) = \{(0, 1), (0.3, 0.7), (0.5, 0.5), (0.7, 0.3), (1, 0)\}$ . Note (1, 0) reduces to standard single-metric optimization of e.g. BLEU.
2. Proposed Pareto approach (PMO-PRO).

Evaluation of multi-objective problems can be tricky because there is no single figure-of-merit. We thus adopted the following methodology: We run both methods 5 times (i.e. using the 5 different  $(p_1, p_2)$  setting each time) and  $I = 20$  iterations each. For each method, this generates  $5 \times 20 = 100$  results, and we plot the Pareto Frontier of these points in a 2-dimensional metric space (e.g. see Figure 2). A method is deemed better if its final Pareto Frontier curve is strictly dominating the other. We report devset results here; testset trends are similar but not included due to space constraints.<sup>7</sup>

<sup>5</sup>from [www.kecl.ntt.co.jp/icl/lirg/ribes](http://www.kecl.ntt.co.jp/icl/lirg/ribes)

<sup>6</sup>from [www.umd.edu/~snoover/tercom](http://www.umd.edu/~snoover/tercom)

<sup>7</sup>An aside: For comparing optimization methods, we believe devset comparison is preferable to testset since data mismatch may confound results. If one worries about generalization, we advocate to *re-decode* the devset with final weights and evaluate its 1-best output (which is done here). This is preferable to simply reporting the achieved scores on devset N-best (as done in some open-source scripts) since the learned weight may pick out good hypotheses in the N-best but perform poorly when re-decoding the same devset. The *re-decode devset* approach avoids being overly optimistic while accurately measuring optimization performance.

	Train	Devset	#Feat	Metrics
PubMed	0.2M	2k	14	BLEU, RIBES
NIST	7M	1.6k	8	BLEU, NTER

Table 1: Task characteristics: #sentences in Train/Dev, # of features, and metrics used. Our MT models are trained with standard phrase-based Moses software (Koehn and others, 2007), with IBM M4 alignments, 4gram SRILM, lexical ordering for PubMed and distance ordering for the NIST system. The decoder generates 50-best lists each iteration. We use SVMRank (Joachims, 2006) as optimization subroutine for PRO, which efficiently handle all pairwise samples without the need for sampling.

## 4.2 Results

Figures 2 and 3 show the results for PubMed and NIST, respectively. A method is better if its Pareto Frontier lies more towards the upper-right hand corner of the graph. Our observations are:

1. PMO-PRO generally outperforms Linear-Combination with any setting of  $(p_1, p_2)$ . The Pareto Frontier of PMO-PRO dominates that of Linear-Combination. This implies PMO is effective in optimizing towards Pareto hypotheses.
2. For both methods, trading-off between metrics is necessary. For example in PubMed, the designer would need to make a choice between picking the best weight according to BLEU (BLEU=.265,RIBES=.665) vs. another weight with higher RIBES but poorer BLEU, e.g. (.255,.675). Nevertheless, both the PMO and Linear-Combination with various  $(p_1, p_2)$  samples this joint-objective space broadly.
3. Interestingly, a multi-objective approach can sometimes outperform a single-objective optimizer in its own metric. In Figure 2, single-objective PRO focusing on optimizing RIBES only achieves 0.68, but PMO-PRO using both BLEU and RIBES outperforms with 0.685.

The third observation relates to the issue of *metric tunability* (Liu et al., 2011). We found that RIBES can be difficult to tune directly. It is an extremely non-smooth objective with many local optima—slight changes in word ordering causes large changes in RIBES. So the best way to improve RIBES is to

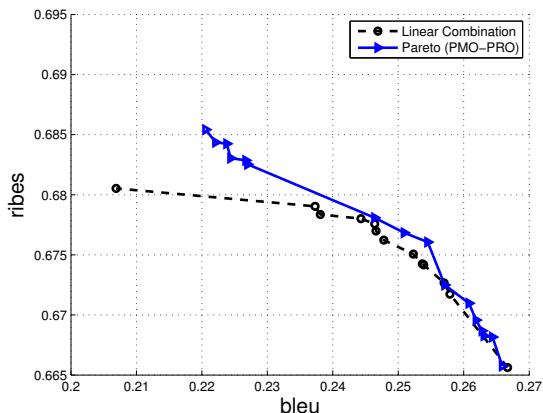


Figure 2: PubMed Results. The curve represents the Pareto Frontier of all results collected after multiple runs.

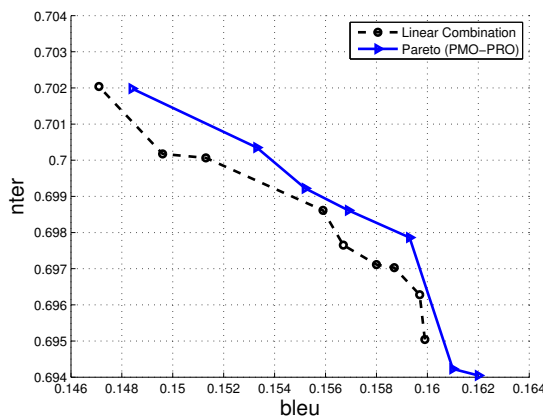


Figure 3: NIST Results

not to optimize it directly, but jointly with a more tunable metric BLEU. The learning curve in Figure 4 show that single-objective optimization of RIBES quickly falls into local optimum (at iteration 3) whereas PMO can zigzag and sacrifice RIBES in intermediate iterations (e.g. iteration 2, 15) leading to a stronger result ultimately. The reason is the diversity of solutions provided by the Pareto Frontier. This finding suggests that multi-objective approaches may be preferred, especially when dealing with new metrics that may be difficult to tune.

## 4.3 Additional Analysis and Discussions

**What is the training time?** The Pareto approach does not add much overhead to PMO-PRO. While FindParetoFrontier scales quadratically by size of N-best list, Figure 5 shows that the runtime is triv-

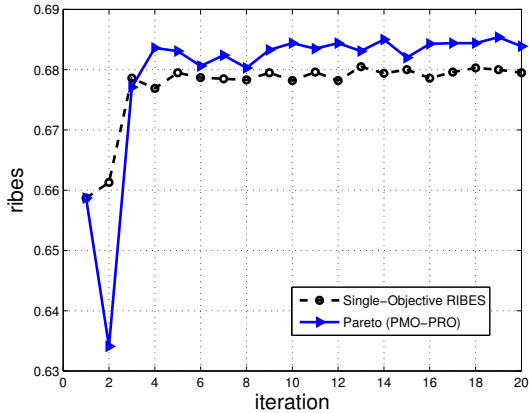


Figure 4: Learning Curve on RIBES: comparing single-objective optimization and PMO.

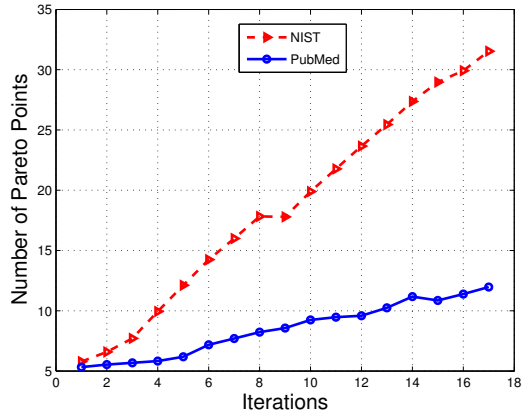


Figure 6: Average number of Pareto points

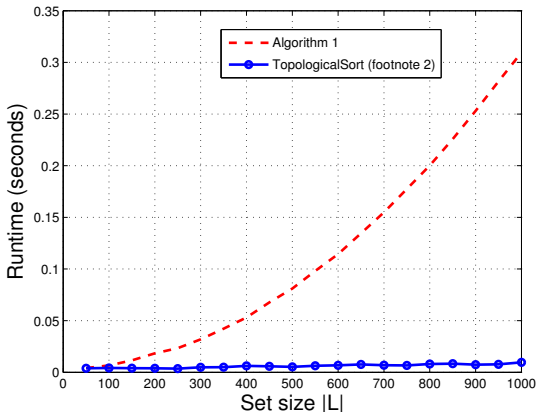


Figure 5: Avg. runtime per sentence of FindPareto

ial (0.3 seconds for 1000-best). Table 2 shows the time usage breakdown in different iterations for PubMed. We see it is mostly dominated by decoding time (constant per iteration at 40 minutes on single 3.33GHz processor). At later iterations, Opt takes more time due to larger file I/O in SVMRank. Note Decode and Pareto can be “embarrassingly parallelized.”

Iter	Time	Decode (line 5)	Pareto (line 7)	Opt (line 10)	Misc. (line 6,8)
1	47m	85%	1%	1%	13%
10	62m	67%	6%	8%	19%
20	91m	47%	15%	22%	16%

Table 2: Training time usage in PMO-PRO (Algo 2).

**How many Pareto points?** The number of pareto

hypotheses gives a rough indication of the diversity of hypotheses that can be exploited by PMO. Figure 6 shows that this number increases gradually per iteration. This perhaps gives PMO-PRO more directions for optimizing around potential local optimal. Nevertheless, we note that tens of Pareto points is far few compared to the large size of N-best lists used at later iterations of PMO-PRO. This may explain why the differences between methods in Figure 3 are not more substantial. Theoretically, the number will eventually level off as it gets increasingly harder to generate new Pareto points in a crowded space (Bentley et al., 1978).

**Practical recommendation:** We present the Pareto approach as a way to agnostically optimize multiple metrics jointly. However, in practice, one may have intuitions about metric tradeoffs even if one cannot specify  $\{p_k\}$ . For example, we might believe that approximately 1-point BLEU degradation is acceptable only if RIBES improves by at least 3-points. In this case, we recommend the following trick: Set up a multi-objective problem where one metric is BLEU and the other is  $3/4\text{BLEU} + 1/4\text{RIBES}$ . This encourages PMO to explore the joint metric space but avoid solutions that sacrifice too much BLEU, and should also outperform Linear Combination that searches only on the  $(3/4, 1/4)$  direction.

## 5 Related Work

Multi-objective optimization for MT is a relatively new area. Linear-combination of BLEU/TER is

the most common technique (Zaidan, 2009), sometimes achieving good results in evaluation campaigns (Dyer et al., 2009). As far as we know, the only work that directly proposes a multi-objective technique is (He and Way, 2009), which modifies MERT to optimize a single metric subject to the constraint that it does not degrade others. These approaches all require some setting of constraint strength or combination weights  $\{p_k\}$ . Recent work in MT evaluation has examined combining metrics using machine learning for better correlation with human judgments (Liu and Gildea, 2007; Albrecht and Hwa, 2007; Gimnez and Màrquez, 2008) and may give insights for setting  $\{p_k\}$ . We view our Pareto-based approach as orthogonal to these efforts.

The tunability of metrics is a problem that is gaining recognition (Liu et al., 2011). If a good evaluation metric could not be used for tuning, it would be a pity. The Tunable Metrics task at WMT2011 concluded that BLEU is still the easiest to tune (Callison-Burch et al., 2011). (Mauser et al., 2008; Cer et al., 2010) report similar observations, in addition citing WER being difficult and BLEU-TER being amenable. One unsolved question is whether metric tunability is a problem inherent to the metric only, or depends also on the underlying optimization algorithm. Our positive results with PMO suggest that the choice of optimization algorithm can help.

Multi-objective ideas are being explored in other NLP areas. (Spitkovsky et al., 2011) describe a technique that alternates between hard and soft EM objectives in order to achieve better local optimum in grammar induction. (Hall et al., 2011) investigates joint optimization of a supervised parsing objective and some extrinsic objectives based on downstream applications. (Agarwal et al., 2011) considers using multiple signals (of varying quality) from online users to train recommendation models. (Eisner and Daumé III, 2011) trades off speed and accuracy of a parser with reinforcement learning. None of the techniques in NLP use Pareto concepts, however.

## 6 Opportunities and Limitations

We introduce a new approach (PMO) for training MT systems on multiple metrics. Leveraging the diverse perspectives of different evaluation metrics has the potential to improve overall quality. Based

on Pareto Optimality, PMO is easy to implement and achieves better solutions compared to linear-combination baselines, for *any setting* of combination weights. Further we observe that multi-objective approaches can be helpful for optimizing difficult-to-tune metrics; this is beneficial for quickly introducing new metrics developed in MT evaluation into MT optimization, especially when good  $\{p_k\}$  are not yet known. We conclude by drawing attention to some limitations and opportunities raised by this work:

**Limitations:** (1) The performance of PMO is limited by the size of the Pareto set. Small N-best lists lead to sparsely-sampled Pareto Frontiers, and a much better approach would be to enlarge the hypothesis space using lattices (Macherey et al., 2008). How to compute Pareto points directly from lattices is an interesting open research question. (2) The binary distinction between pareto vs. non-pareto points ignores the fact that 2nd-place non-pareto points may also lead to good practical solutions. A better approach may be to adopt a *graded* definition of Pareto optimality as done in some multi-objective works (Deb et al., 2002). (3) A robust evaluation methodology that enables significance testing for multi-objective problems is sorely needed. This will make it possible to compare multi-objective methods on more than 2 metrics. We also need to follow up with human evaluation.

**Opportunities:** (1) There is still much we do not understand about metric tunability; we can learn much by looking at joint metric-spaces and examining how new metrics correlate with established ones. (2) Pareto is just one approach among many in multi-objective optimization. A wealth of methods are available (Marler and Arora, 2004) and more experimentation in this space will definitely lead to new insights. (3) Finally, it would be interesting to explore other creative uses of multiple-objectives in MT beyond multiple metrics. For example: Can we learn to translate faster while sacrificing little on accuracy? Can we learn to jointly optimize cascaded systems, such as as speech translation or pivot translation? Life is full of multiple competing objectives.

## Acknowledgments

We thank the reviewers for insightful feedback.

## References

- Deepak Agarwal, Bee-Chung Chen, Pradheep Elango, and Xuanhui Wang. 2011. Click shaping to optimize multiple objectives. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 132–140, New York, NY, USA. ACM.
- J. Albrecht and R. Hwa. 2007. A re-examination of machine learning approaches for sentence-level mt evaluation. In *ACL*.
- J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. 1978. On the average number of maxima in a set of vectors and applications. *Journal of the Association for Computing Machinery (JACM)*, 25(4).
- Alexandra Birch, Phil Blunsom, and Miles Osborne. 2010. Metrics for MT evaluation: Evaluating reordering. *Machine Translation*, 24(1).
- S. Börzsönyi, D. Kossmann, and K. Stocker. 2001. The skyline operator. In *Proceedings of the 17th International Conference on Data Engineering (ICDE)*.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Daniel Cer, Christopher Manning, and Daniel Jurafsky. 2010. The best lexical metric for phrase-based statistical MT system optimization. In *NAACL HLT*.
- David Chiang, Wei Wang, and Kevin Knight. 2009. 11,001 new features for statistical machine translation. In *NAACL*.
- Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7.
- Kalyanmoy Deb, Amrit Pratap, Sammer Agarwal, and T. Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2).
- Chris Dyer, Hendra Setiawan, Yuval Marton, and Philip Resnik. 2009. The university of maryland statistical machine translation system for the fourth workshop on machine translation. In *Proc. of the Fourth Workshop on Machine Translation*.
- Jason Eisner and Hal Daumé III. 2011. Learning speed-accuracy tradeoffs in nondeterministic inference algorithms. In *COST: NIPS 2011 Workshop on Computational Trade-offs in Statistical Learning*.
- Jesús Gimnez and Lluís Màrquez. 2008. Heterogeneous automatic mt evaluation through non-parametric metric combinations. In *ICJNLP*.
- Parke Godfrey, Ryan Shipley, and Jarek Gyrz. 2007. Algorithms and analyses for maximal vector computation. *VLDB Journal*, 16.
- Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the ntcir-9 workshop. In *Proceedings of the NTCIR-9 Workshop Meeting*.
- Keith Hall, Ryan McDonald, Jason Katz-Brown, and Michael Ringgaard. 2011. Training dependency parsers by jointly optimizing multiple objectives. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1489–1499, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Yifan He and Andy Way. 2009. Improving the objective function in minimum error rate training. In *MT Summit*.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- H. Isozaki, T. Hirao, K. Duh, K. Sudoh, and H. Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *EMNLP*.
- T. Joachims. 2006. Training linear SVMs in linear time. In *KDD*.
- P. Koehn et al. 2007. Moses: open source toolkit for statistical machine translation. In *ACL*.
- A. Lavie and A. Agarwal. 2007. METEOR: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Workshop on Statistical Machine Translation*.
- P. Liang, A. Bouchard-Cote, D. Klein, and B. Taskar. 2006. An end-to-end discriminative approach to machine translation. In *ACL*.
- Ding Liu and Daniel Gildea. 2007. Source-language features and maximum correlation training for machine translation evaluation. In *NAACL*.
- Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2011. Better evaluation metrics lead to better machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *EMNLP*.
- R. T. Marler and J. S. Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26.
- Arne Mauser, Saša Hasan, and Hermann Ney. 2008. Automatic evaluation measures for statistical machine

- translation system optimization. In *International Conference on Language Resources and Evaluation*, Marrakech, Morocco, May.
- Kaisa Miettinen. 1998. *Nonlinear Multiobjective Optimization*. Springer.
- J.A. Nelder and R. Mead. 1965. The downhill simplex method. *Computer Journal*, 7(308).
- Franz Och. 2003. Minimum error rate training in statistical machine translation. In *ACL*.
- Karolina Owczarzak, Josef van Genabith, and Andy Way. 2007. Labelled dependencies in machine translation evaluation. In *Proceedings of the Second Workshop on Statistical Machine Translation*.
- Sebastian Pado, Daniel Cer, Michel Galley, Dan Jurafsky, and Christopher D. Manning. 2009. Measuring machine translation quality as semantic equivalence: A metric based on entailment features. *Machine Translation*, 23(2-3).
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*.
- Vilfredo Pareto. 1906. *Manuale di Economica Politica*, (Translated into English by A.S. Schwier as *Manual of Political Economy*, 1971). Societa Editrice Libreria, Milan.
- Michael Paul. 2010. Overview of the iwslt 2010 evaluation campaign. In *IWSLT*.
- Yoshikazu Sawaragi, Hirotaka Nakayama, and Tetsuzo Tanino, editors. 1985. *Theory of Multiobjective Optimization*. Academic Press.
- M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *AMTA*.
- Valentin I. Spitzkovsky, Hiyan Alshawi, and Daniel Jurafsky. 2011. Lateen em: Unsupervised training with multiple objectives, applied to dependency grammar induction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1269–1280, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Omar Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. In *The Prague Bulletin of Mathematical Linguistics*.

# Joint Feature Selection in Distributed Stochastic Learning for Large-Scale Discriminative Training in SMT

**Patrick Simianer** and **Stefan Riezler**

Department of Computational Linguistics  
Heidelberg University  
69120 Heidelberg, Germany

{simianer, riezler}@cl.uni-heidelberg.de

**Chris Dyer**

Language Technologies Institute  
Carnegie Mellon University  
Pittsburgh, PA, 15213, USA

cdyer@cs.cmu.edu

## Abstract

With a few exceptions, discriminative training in statistical machine translation (SMT) has been content with tuning weights for large feature sets on small development data. Evidence from machine learning indicates that increasing the training sample size results in better prediction. The goal of this paper is to show that this common wisdom can also be brought to bear upon SMT. We deploy local features for SCFG-based SMT that can be read off from rules at runtime, and present a learning algorithm that applies  $\ell_1/\ell_2$  regularization for joint feature selection over distributed stochastic learning processes. We present experiments on learning on 1.5 million training sentences, and show significant improvements over tuning discriminative models on small development sets.

## 1 Introduction

The standard SMT training pipeline combines scores from large count-based translation models and language models with a few other features and tunes these using the well-understood line-search technique for error minimization of Och (2003). If only a handful of dense features need to be tuned, minimum error rate training can be done on small tuning sets and is hard to beat in terms of accuracy and efficiency. In contrast, the promise of large-scale discriminative training for SMT is to scale to arbitrary types and numbers of features and to provide sufficient statistical support by parameter estimation on large sample sizes. Features may be lexicalized and sparse, non-local and overlapping, or

be designed to generalize beyond surface statistics by incorporating part-of-speech or syntactic labels. The modeler's goals might be to identify complex properties of translations, or to counter errors of pre-trained translation models and language models by explicitly down-weighting translations that exhibit certain undesired properties. Various approaches to feature engineering for discriminative models have been presented (see Section 2), however, with a few exceptions, discriminative learning in SMT has been confined to training on small tuning sets of a few thousand examples. This contradicts theoretical and practical evidence from machine learning that suggests that larger training samples should be beneficial to improve prediction also in SMT. Why is this?

One possible reason why discriminative SMT has mostly been content with small tuning sets lies in the particular design of the features themselves. For example, the features introduced by Chiang et al. (2008) and Chiang et al. (2009) for an SCFG model for Chinese/English translation are of two types: The first type explicitly counters overestimates of rule counts, or rules with bad overlap points, bad rewrites, or with undesired insertions of target-side terminals. These features are specified in hand-crafted lists based on a thorough analysis of a tuning set. Such finely hand-crafted features will find sufficient statistical support on a few thousand examples and thus do not benefit from larger training sets. The second type of features deploys external information such as syntactic parses or word alignments to penalize bad reorderings or undesired translations of phrases that cross syntactic constraints. At large scale, extraction of such features quickly becomes

- (1)  $X \rightarrow X_1 \text{ hat } X_2 \text{ versprochen, } X_1 \text{ promised } X_2$
- (2)  $X \rightarrow X_1 \text{ hat mir } X_2 \text{ versprochen, } X_1 \text{ promised me } X_2$
- (3)  $X \rightarrow X_1 \text{ versprach } X_2, X_1 \text{ promised } X_2$

Figure 1: SCFG rules for translation.

infeasible because of costly generation and storage of linguistic annotations. Another possible reason why large training data did not yet show the expected improvements in discriminative SMT is a special overfitting problem of current popular online learning techniques. This is due to stochastic learning on a per-example basis where a weight update on a misclassified example may apply only to a small fraction of data that have been seen before. Thus many features will not generalize well beyond the training examples on which they were introduced.

The goal of this paper is to investigate if and how it is possible to benefit from scaling discriminative training for SMT to large training sets. We deploy generic features for SCFG-based SMT that can efficiently be read off from rules at runtime. Such features include rule ids, rule-local n-grams, or types of rule shapes. Another crucial ingredient of our approach is a combination of parallelized stochastic learning with feature selection inspired by multi-task learning. The simple but effective idea is to randomly divide training data into evenly sized shards, use stochastic learning on each shard in parallel, while performing  $\ell_1/\ell_2$  regularization for joint feature selection on the shards after each epoch, before starting a new epoch with a reduced feature vector averaged across shards. Iterative feature selection procedure is the key to both efficiency and improved prediction: Without interleaving parallelized stochastic learning with feature selection our largest experiments would not be feasible. Selecting features jointly across shards and averaging does counter the overfitting effect that is inherent to stochastic updating. Our resulting models are learned on large data sets, but they are small and outperform models that tune feature sets of various sizes on small development sets. Our software is freely available as a part of the `cdec`<sup>1</sup> framework.

<sup>1</sup><https://github.com/redpony/cdec>

## 2 Related Work

The great promise of discriminative training for SMT is the possibility to design arbitrarily expressive, complex, or overlapping features in great numbers. The focus of many approaches thus has been on feature engineering and on adaptations of machine learning algorithms to the special case of SMT (where gold standard rankings have to be created automatically). Examples for adapted algorithms include Maximum-Entropy Models (Och and Ney, 2002; Blunsom et al., 2008), Pairwise Ranking Perceptrons (Shen et al., 2004; Watanabe et al., 2006; Hopkins and May, 2011), Structured Perceptrons (Liang et al., 2006a), Boosting (Duh and Kirchhoff, 2008; Wellington et al., 2009), Structured SVMs (Tillmann and Zhang, 2006; Hayashi et al., 2009), MIRA (Watanabe et al., 2007; Chiang et al., 2008; Chiang et al., 2009), and others. Adaptations of the loss functions underlying such algorithms to SMT have recently been described as particular forms of ramp loss optimization (McAllester and Keshet, 2011; Gimpel and Smith, 2012).

All approaches have been shown to scale to large feature sets and all include some kind of regularization method. However, most approaches have been confined to training on small tuning sets. Exceptions where discriminative SMT has been used on large training data are Liang et al. (2006a) who trained 1.5 million features on 67,000 sentences, Blunsom et al. (2008) who trained 7.8 million rules on 100,000 sentences, or Tillmann and Zhang (2006) who used 230,000 sentences for training.

Our approach is inspired by Duh et al. (2010) who applied multi-task learning for improved generalization in n-best reranking. In contrast to our work, Duh et al. (2010) did not incorporate multi-task learning into distributed learning, but defined tasks as n-best lists, nor did they develop new algorithms, but used off-the-shelf multi-task tools.

## 3 Local Features for Synchronous CFGs

The work described in this paper is based on the SMT framework of hierarchical phrase-based translation (Chiang, 2005; Chiang, 2007). Translation rules are extracted from word-aligned parallel sentences and can be seen as productions of a synchronous CFG. Examples are rules like (1)-(3)



shown in Figure 1. Local features are designed to be readable directly off the rule at decoding time. We use three rule templates in our work:

**Rule identifiers:** These features identify each rule by a unique identifier. Such features correspond to the relative frequencies of rewrites rules used in standard models.

**Rule n-grams:** These features identify n-grams of consecutive items in a rule. We use bigrams on source-sides of rules. Such features identify possible source side phrases and thus can give preference to rules including them.<sup>2</sup>

**Rule shape:** These features are indicators that abstract away from lexical items to templates that identify the location of sequences of terminal symbols in relation to non-terminal symbols, on both the source- and target-sides of each rule used. For example, both rules (1) and (2) map to the same indicator, namely that a rule is being used that consists of a (NT, term\*, NT, term\*) pattern on its source side, and an (NT, term\*, NT) pattern on its target side. Rule (3) maps to a different template, that of (NT, term\*, NT) on source and target sides.

#### 4 Joint Feature Selection in Distributed Stochastic Learning

The following discussion of learning methods is based on pairwise ranking in a Stochastic Gradient Descent (SGD) framework. The resulting algorithms can be seen as variants of the perceptron algorithm. Let each translation candidate be represented by a feature vector  $\mathbf{x} \in \mathbb{R}^D$  where preference pairs for training are prepared by sorting translations according to smoothed sentence-wise BLEU score (Liang et al., 2006a) against the reference. For a preference pair  $\mathbf{x}_j = (\mathbf{x}_j^{(1)}, \mathbf{x}_j^{(2)})$  where  $\mathbf{x}_j^{(1)}$  is preferred over  $\mathbf{x}_j^{(2)}$ , and  $\bar{\mathbf{x}}_j = \mathbf{x}_j^{(1)} - \mathbf{x}_j^{(2)}$ , we consider the following hinge loss-type objective function:

$$l_j(\mathbf{w}) = (-\langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle)_+$$

where  $(a)_+ = \max(0, a)$ ,  $\mathbf{w} \in \mathbb{R}^D$  is a weight vector, and  $\langle \cdot, \cdot \rangle$  denotes the standard vector dot product. Instantiating SGD to the following stochastic

<sup>2</sup>Similar ‘‘monolingual parse features’’ have been used in Dyer et al. (2011).

subgradient leads to the perceptron algorithm for pairwise ranking<sup>3</sup> (Shen and Joshi, 2005):

$$\nabla l_j(\mathbf{w}) = \begin{cases} -\bar{\mathbf{x}}_j & \text{if } \langle \mathbf{w}, \bar{\mathbf{x}}_j \rangle \leq 0, \\ 0 & \text{else.} \end{cases}$$

Our baseline algorithm 1 (SDG) scales pairwise ranking to large scale scenarios. The algorithm takes an average over the final weight updates of each epoch instead of keeping a record of all weight updates for final averaging (Collins, 2002) or for voting (Freund and Schapire, 1999).

---

#### Algorithm 1 SGD: int $I, T$ , float $\eta$

---

```

Initialize  $\mathbf{w}_{0,0,0} \leftarrow \mathbf{0}$ .
for epochs  $t \leftarrow 0 \dots T - 1$ : do
  for all  $i \in \{0 \dots I - 1\}$ : do
    Decode  $i^{\text{th}}$  input with  $\mathbf{w}_{t,i,0}$ .
    for all pairs  $x_j, j \in \{0 \dots P - 1\}$ : do
       $\mathbf{w}_{t,i,j+1} \leftarrow \mathbf{w}_{t,i,j} - \eta \nabla l_j(\mathbf{w}_{t,i,j})$ 
    end for
     $\mathbf{w}_{t,i+1,0} \leftarrow \mathbf{w}_{t,i,P}$ 
  end for
   $\mathbf{w}_{t+1,0,0} \leftarrow \mathbf{w}_{t,I,0}$ 
end for
return  $\frac{1}{T} \sum_{t=1}^T \mathbf{w}_{t,0,0}$ 

```

---

While stochastic learning exhibits a runtime behavior that is linear in sample size (Bottou, 2004), very large datasets can make sequential processing infeasible. Algorithm 2 (MixSGD) addresses this problem by parallelization in the framework of MapReduce (Dean and Ghemawat, 2004).

---

#### Algorithm 2 MixSGD: int $I, T, Z$ , float $\eta$

---

```

Partition data into  $Z$  shards, each of size  $S \leftarrow I/Z$ ;
distribute to machines.
for all shards  $z \in \{1 \dots Z\}$ : parallel do
  Initialize  $\mathbf{w}_{z,0,0,0} \leftarrow \mathbf{0}$ .
  for epochs  $t \leftarrow 0 \dots T - 1$ : do
    for all  $i \in \{0 \dots S - 1\}$ : do
      Decode  $i^{\text{th}}$  input with  $\mathbf{w}_{z,t,i,0}$ .
      for all pairs  $x_j, j \in \{0 \dots P - 1\}$ : do
         $\mathbf{w}_{z,t,i,j+1} \leftarrow \mathbf{w}_{z,t,i,j} - \eta \nabla l_j(\mathbf{w}_{z,t,i,j})$ 
      end for
       $\mathbf{w}_{z,t,i+1,0} \leftarrow \mathbf{w}_{z,t,i,P}$ 
    end for
     $\mathbf{w}_{z,t+1,0,0} \leftarrow \mathbf{w}_{z,t,S,0}$ 
  end for
end for
Collect final weights from each machine,
return  $\frac{1}{Z} \sum_{z=1}^Z \left( \frac{1}{T} \sum_{t=1}^T \mathbf{w}_{z,t,0,0} \right)$ .

```

---

<sup>3</sup>Other loss functions lead to stochastic versions of SVMs (Collobert and Bengio, 2004; Shalev-Shwartz et al., 2007; Chapelle and Keerthi, 2010).

Algorithm 2 is a variant of the SimuParallelSGD algorithm of Zinkevich et al. (2010) or equivalently of the parameter mixing algorithm of McDonald et al. (2010). The key idea of algorithm 2 is to partition the data into disjoint shards, then train SGD on each shard in parallel, and after training mix the final parameters from each shard by averaging. The algorithm requires no communication between machines until the end.

McDonald et al. (2010) also present an iterative mixing algorithm where weights are mixed from each shard after training a single epoch of the perceptron in parallel on each shard. The mixed weight vector is re-sent to each shard to start another epoch of training in parallel on each shard. This algorithm corresponds to our algorithm 3 (IterMixSGD).

---

**Algorithm 3** IterMixSGD: int  $I, T, Z$ , float  $\eta$

---

Partition data into  $Z$  shards, each of size  $S \leftarrow I/Z$ ; distribute to machines.

Initialize  $\mathbf{v} \leftarrow \mathbf{0}$ .

```

for epochs  $t \leftarrow 0 \dots T - 1$ : do
  for all shards  $z \in \{1 \dots Z\}$ : parallel do
     $\mathbf{w}_{z,t,0,0} \leftarrow \mathbf{v}$ 
    for all  $i \in \{0 \dots S - 1\}$ : do
      Decode  $i^{\text{th}}$  input with  $\mathbf{w}_{z,t,i,0}$ .
      for all pairs  $x_j, j \in \{0 \dots P - 1\}$ : do
         $\mathbf{w}_{z,t,i,j+1} \leftarrow \mathbf{w}_{z,t,i,j} - \eta \nabla l_j(\mathbf{w}_{z,t,i,j})$ 
      end for
       $\mathbf{w}_{z,t,i+1,0} \leftarrow \mathbf{w}_{z,t,i,P}$ 
    end for
  end for
  Collect weights  $\mathbf{v} \leftarrow \frac{1}{Z} \sum_{z=1}^Z \mathbf{w}_{z,t,S,0}$ .

```

```

end for
return  $\mathbf{v}$ 

```

---

Parameter mixing by averaging will help to ease the feature sparsity problem, however, keeping feature vectors on the scale of several million features in memory can be prohibitive. If network latency is a bottleneck, the increased amount of information sent across the network after each epoch may be a further problem.

Our algorithm 4 (IterSelSGD) introduces feature selection into distributed learning for increased efficiency and as a more radical measure against overfitting. The key idea is to view shards as tasks, and to apply methods for joint feature selection from multi-task learning to achieve small sets of features that are useful across all tasks or shards. Our algorithm represents weights in a  $Z$ -by- $D$  matrix  $\mathbf{W} = [\mathbf{w}_{z_1} | \dots | \mathbf{w}_{z_Z}]^T$  of stacked  $D$ -dimensional weight

vectors across  $Z$  shards. We compute the  $\ell_2$  norm of the weights in each feature column, sort features by this value, and keep  $K$  features in the model. This feature selection procedure is done after each epoch. Reduced weight vectors are mixed and the result is re-sent to each shard to start another epoch of parallel training on each shard.

---

**Algorithm 4** IterSelSGD: int  $I, T, Z, K$ , float  $\eta$

---

Partition data into  $Z$  shards, each of size  $S = I/Z$ ; distribute to machines.

Initialize  $\mathbf{v} \leftarrow \mathbf{0}$ .

```

for epochs  $t \leftarrow 0 \dots T - 1$ : do
  for all shards  $z \in \{1 \dots Z\}$ : parallel do
     $\mathbf{w}_{z,t,0,0} \leftarrow \mathbf{v}$ 
    for all  $i \in \{0 \dots S - 1\}$ : do
      Decode  $i^{\text{th}}$  input with  $\mathbf{w}_{z,t,i,0}$ .
      for all pairs  $x_j, j \in \{0 \dots P - 1\}$ : do
         $\mathbf{w}_{z,t,i,j+1} \leftarrow \mathbf{w}_{z,t,i,j} - \eta \nabla l_j(\mathbf{w}_{z,t,i,j})$ 
      end for
       $\mathbf{w}_{z,t,i+1,0} \leftarrow \mathbf{w}_{z,t,i,P}$ 
    end for
  end for
  Collect/stack weights  $\mathbf{W} \leftarrow [\mathbf{w}_{1,t,S,0} | \dots | \mathbf{w}_{Z,t,S,0}]^T$ 
  Select top  $K$  feature columns of  $\mathbf{W}$  by  $\ell_2$  norm and
  for  $k \leftarrow 1 \dots K$  do
     $\mathbf{v}[k] = \frac{1}{Z} \sum_{z=1}^Z \mathbf{W}[z][k]$ .
  end for
end for
return  $\mathbf{v}$ 

```

---

This algorithm can be seen as an instance of  $\ell_1/\ell_2$  regularization as follows: Let  $w_d$  be the  $d$ th column vector of  $\mathbf{W}$ , representing the weights for the  $d$ th feature across tasks/shards.  $\ell_1/\ell_2$  regularization penalizes weights  $\mathbf{W}$  by the weighted  $\ell_1/\ell_2$  norm

$$\lambda \|\mathbf{W}\|_{1,2} = \lambda \sum_{d=1}^D \|w_d\|_2.$$

Each  $\ell_2$  norm of a weight column represents the relevance of the corresponding feature across tasks/shards. The  $\ell_1$  sum of the  $\ell_2$  norms enforces a selection among features based on these norms. Consider for example the two 5-feature, 3-task weight matrices in Figure 2. Assuming the same loss for both matrices, the right-hand side matrix is preferred because of a smaller  $\ell_1/\ell_2$  norm (12 instead of 18). This matrix shares features across tasks which leads to larger  $\ell_2$  norms for some columns (here  $\|w_1\|_2$  and  $\|w_2\|_2$ ) and forces other columns to zero. This results in shrinking the matrix to those features that are useful across all tasks.

		$w_1$	$w_2$	$w_3$	$w_4$	$w_5$		$w_1$	$w_2$	$w_3$	$w_4$	$w_5$
	$\mathbf{w}_{z_1}$	[	6	4	0	0		[	6	4	0	0
	$\mathbf{w}_{z_2}$	[	0	0	3	0		[	3	0	0	0
	$\mathbf{w}_{z_3}$	[	0	0	0	2		[	2	3	0	0
column $\ell_2$ norm:			6	4	3	2			7	5	0	0
$\ell_1$ sum:						$\Rightarrow$ 18						$\Rightarrow$ 12

Figure 2:  $\ell_1/\ell_2$  regularization enforcing feature selection.

Our algorithm is related to Obozinski et al. (2010)’s approach to  $\ell_1/\ell_2$  regularization where feature columns are incrementally selected based on the  $\ell_2$  norms of the gradient vectors corresponding to feature columns. Their algorithm is itself an extension of gradient-based feature selection based on the  $\ell_1$  norm, e.g., Perkins et al. (2003).<sup>4</sup> In contrast to these approaches we approximate the gradient by using the weights given by the ranking algorithm itself. This relates our work to weight-based recursive feature elimination (RFE) (Lal et al., 2006). Furthermore, algorithm 4 performs feature selection based on a choice of meta-parameter of  $K$  features instead of by thresholding a regularization meta-parameter  $\lambda$ , however, these techniques are equivalent and can be transformed into each other.

## 5 Experiments

### 5.1 Data, Systems, Experiment Settings

The datasets used in our experiments are versions of the News Commentary (*nc*), News Crawl (*crawl*) and Europarl (*ep*) corpora described in Table 1. The translation direction is German-to-English.

The SMT framework used in our experiments is hierarchical phrase-based translation (Chiang, 2007). We use the `cdec` decoder<sup>5</sup> (Dyer et al., 2010) and induce SCFG grammars from two sets of symmetrized alignments using the method described by Chiang (2007). All data was tokenized and lowercased; German compounds were split (Dyer, 2009). For word alignment of the news-commentary data, we used GIZA++ (Och and Ney, 2000); for aligning the Europarl data, we used the Berkeley aligner (Liang et al., 2006b). Before training, we collect all the grammar rules necessary to

translate each individual sentence into separate files (so-called per-sentence grammars) (Lopez, 2007). When decoding, `cdec` loads the appropriate file immediately prior to translation of the sentence. The computational overhead is minimal compared to the expense of decoding. Also, deploying disk space instead of memory fits perfectly into the MapReduce framework we are working in. Furthermore, the extraction of grammars for training is done in a leave-one-out fashion (Zollmann and Sima’an, 2005) where rules are extracted for a parallel sentence pair only if the same rules are found in other sentences of the corpus as well.

3-gram (news-commentary) and 5-gram (Europarl) language models are trained on the data described in Table 1, using the SRILM toolkit (Stolcke, 2002) and binarized for efficient querying using kenlm (Heafield, 2011). For the 5-gram language models, we replaced every word in the lm training data with `<unk>` that did not appear in the English part of the parallel training data to build an open vocabulary language model.

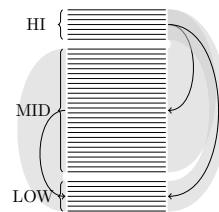


Figure 3: Multipartite pairwise ranking.

Training data for discriminative learning are prepared by comparing a 100-best list of translations against a single reference using smoothed per-sentence BLEU (Liang et al., 2006a). From the BLEU-reordered n-best list, translations were put into sets for the top 10% level (HI), the middle 80% level (MID), and the bottom 10% level (LOW). These level sets are used for multipartite ranking

<sup>4</sup>Note that by definition of  $\|\mathbf{W}\|_{1,2}$ , standard  $\ell_1$  regularization is a special case of  $\ell_1/\ell_2$  regularization for a single task.

<sup>5</sup>`cdec` metaparameters were set to a non-terminal span limit of 15 and standard cube pruning with a pop limit of 200.

News Commentary( <i>nc</i> )					
	train- <i>nc</i>	lm-train- <i>nc</i>	dev- <i>nc</i>	devtest- <i>nc</i>	test- <i>nc</i>
Sentences	132,753	180,657	1057	1064	2007
Tokens <i>de</i>	3,530,907	–	27,782	28,415	53,989
Tokens <i>en</i>	3,293,363	4,394,428	26,098	26,219	50,443
Rule Count	14,350,552 (1G)	–	2,322,912	2,320,264	3,274,771
Europarl( <i>ep</i> )					
	train- <i>ep</i>	lm-train- <i>ep</i>	dev- <i>ep</i>	devtest- <i>ep</i>	test- <i>ep</i>
Sentences	1,655,238	2,015,440	2000	2000	2000
Tokens <i>de</i>	45,293,925	–	57,723	56,783	59,297
Tokens <i>en</i>	45,374,649	54,728,786	58,825	58,100	60,240
Rule Count	203,552,525 (31.5G)	–	17,738,763	17,682,176	18,273,078
News Crawl( <i>crawl</i> )					
			dev- <i>crawl</i>	test- <i>crawl10</i>	test- <i>crawl11</i>
Sentences			2051	2489	3003
Tokens <i>de</i>			49,848	64,301	76,193
Tokens <i>en</i>			49,767	61,925	74,753
Rule Count			9,404,339	11,307,304	12,561,636

Table 1: Overview of data used for train/dev/test. News Commentary (*nc*) and Europarl (*ep*) training data and also News Crawl (*crawl*) dev/test data were taken from the WMT11 translation task (<http://statmt.org/wmt11/translation-task.html>). The dev/test data of *nc* are the sets provided with the WMT07 shared task (<http://statmt.org/wmt07/shared-task.html>). *Ep* dev/test data is from WMT08 shared task (<http://statmt.org/wmt08/shared-task.html>). The numbers in brackets for the rule counts of *ep/nc* training data are total counts of rules in the per-sentence grammars.

where translation pairs are built between the elements in HI-MID, HI-LOW, and MID-LOW, but not between translations inside sets on the same level. This idea is depicted graphically in Figure 3. The intuition is to ensure that good translations are preferred over bad translations without teasing apart small differences.

For evaluation, we used the `mteval-v11b.pl` script to compute lowercased BLEU-4 scores (Papineni et al., 2001). Statistical significance was measured using an Approximate Randomization test (Noreen, 1989; Riezler and Maxwell, 2005).

All experiments for training on dev sets were carried out on a single computer. For grammar extraction and training of the full data set we used a 30 node hadoop Map/Reduce cluster that can handle 300 jobs at once. We split the data into 2290 shards for the *ep* runs and 141 shards for the *nc* runs, each shard holding about 1,000 sentences, which corresponds to the dev set size of the *nc* data set.

## 5.2 Experimental Results

The baseline learner in our experiments is a pairwise ranking perceptron that is used on various features and training data and plugged into various meta-

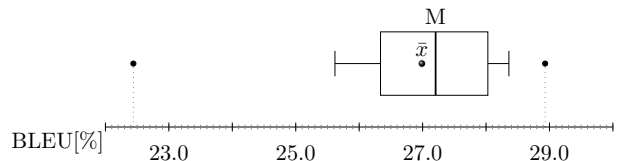


Figure 4: Boxplot of BLEU-4 results for 100 runs of MIRA on news commentary data, depicting median (M), mean ( $\bar{x}$ ), interquartile range (box), standard deviation (whiskers), outliers (end points).

algorithms for distributed processing. The perceptron algorithm itself compares favorably to related learning techniques such as the MIRA adaptation of Chiang et al. (2008). Figure 4 gives a boxplot depicting BLEU-4 results for 100 runs of the MIRA implementation of the `cdec` package, tuned on dev-*nc*, and evaluated on the respective test set test-*nc*.<sup>6</sup> We see a high variance (whiskers denote standard deviations) around a median of 27.2 BLEU and a mean of 27.1 BLEU. The fluctuation of results is due to sampling training examples from the translation hy-

<sup>6</sup>MIRA was used with default meta parameters: 250 hypothesis list to search for oracles, regularization strength  $C = 0.01$  and using 15 passes over the input. It optimized IBM BLEU-4. The initial weight vector was  $\mathbf{0}$ .

Algorithm	Tuning set	Features	#Features	devtest- <i>nc</i>	test- <i>nc</i>
MIRA	dev- <i>nc</i>	default	12	–	27.10
1	dev- <i>nc</i>	default	12	<b>25.88</b>	28.0
	dev- <i>nc</i>	+id	137k	25.53	27.6 <sup>†23</sup>
	dev- <i>nc</i>	+ng	29k	25.82	27.42 <sup>†234</sup>
	dev- <i>nc</i>	+shape	51	25.91	28.1
	dev- <i>nc</i>	+id,ng,shape	180k	25.71	<b>28.15</b> <sup>34</sup>
2	train- <i>nc</i>	default	12	25.73	27.86
	train- <i>nc</i>	+id	4.1M	25.13	27.19 <sup>†134</sup>
	train- <i>nc</i>	+ng	354k	<b>26.09</b>	<b>28.03</b> <sup>134</sup>
	train- <i>nc</i>	+shape	51	26.07	27.91 <sup>3</sup>
	train- <i>nc</i>	+id,ng,shape	4.7M	26.08	27.86 <sup>34</sup>
3	train- <i>nc</i>	default	12	26.09 @2	27.94 <sup>†</sup>
	train- <i>nc</i>	+id	3.4M	26.1 @4	27.97 <sup>†12</sup>
	train- <i>nc</i>	+ng	330k	26.33 @4	28.34 <sup>12</sup>
	train- <i>nc</i>	+shape	51	26.39 @9	28.31 <sup>2</sup>
	train- <i>nc</i>	+id,ng,shape	4.7M	<b>26.42 @9</b>	<b>28.55</b> <sup>124</sup>
4	train- <i>nc</i>	+id	100k	25.91 @7	27.82 <sup>†2</sup>
	train- <i>nc</i>	+ng	100k	26.42 @4	28.37 <sup>†12</sup>
	train- <i>nc</i>	+id,ng,shape	100k	<b>26.8 @8</b>	<b>28.81</b> <sup>123</sup>

Table 2: BLEU-4 results for algorithms 1 (SGD), 2 (MixSGD), 3 (IterMixSDG), and 4 (IterSelSGD) on news-commentary (*nc*) data. Feature groups are 12 dense features (default), rule identifiers (id), rule n-gram (ng), and rule shape (shape). Statistical significance at  $p$ -level  $< 0.05$  of a result difference on the test set to a different algorithm applied to the same feature group is indicated by raised algorithm number.  $\dagger$  indicates statistically significant differences to best result across features groups for same algorithm, indicated in **bold face**. @ indicates the optimal number of epochs chosen on the devtest set.

pergraph as is done in the `cdec` implementation of MIRA. We found similar fluctuations for the `cdec` implementations of PRO (Hopkins and May, 2011) or hypergraph-MERT (Kumar et al., 2009) both of which depend on hypergraph sampling. In contrast, the perceptron is deterministic when started from a zero-vector of weights and achieves favorable 28.0 BLEU on the news-commentary test set. Since we are interested in relative improvements over a stable baseline, we restrict our attention in all following experiments to the perceptron.<sup>7</sup>

Table 2 shows the results of the experimental comparison of the 4 algorithms of Section 4. The

<sup>7</sup>Absolute improvements would be possible, e.g., by using larger language models or by adding news data to the *ep* training set when evaluating on *crawl* test sets (see, e.g., Dyer et al. (2011)), however, this is not the focus of this paper.

default features include 12 dense models defined on SCFG rules;<sup>8</sup> The sparse features are the 3 templates described in Section 3. All feature weights were tuned together using algorithms 1-4. If not indicated otherwise, the perceptron was run for 10 epochs with learning rate  $\eta = 0.0001$ , started at zero weight vector, using deduplicated 100-best lists.

The results on the news-commentary (*nc*) data show that training on the development set does not benefit from adding large feature sets – BLEU result differences between tuning 12 default features

<sup>8</sup>negative log relative frequency  $p(e|f)$ ; log count( $f$ ); log count( $e, f$ ); lexical translation probability  $p(f|e)$  and  $p(e|f)$  (Koehn et al., 2003); indicator variable on singleton phrase  $e$ ; indicator variable on singleton phrase pair  $f, e$ ; word penalty; language model weight; OOV count of language model; number of untranslated words; Hiero glue rules (Chiang, 2007).

Alg.	Tuning set	Features	#Feats	devtest- <i>ep</i>	test- <i>ep</i>	Tuning set	test- <i>crawl10</i>	test- <i>crawl11</i>
1	dev- <i>ep</i>	default	12	25.62	26.42 <sup>†</sup>	dev- <i>crawl</i>	15.39 <sup>†</sup>	14.43 <sup>†</sup>
	dev- <i>ep</i>	+id,ng,shape	300k	<b>27.84</b>	<b>28.37</b>	dev- <i>crawl</i>	<b>17.8</b> <sup>4</sup>	<b>16.83</b> <sup>4</sup>
4	train- <i>ep</i>	+id,ng,shape	100k	<b>28.0</b> @9	<b>28.62</b>	train- <i>ep</i>	<b>19.12</b> <sup>1</sup>	<b>17.33</b> <sup>1</sup>

Table 3: BLEU-4 results for algorithms 1 (SGD) and 4 (IterSelSGD) on Europarl (*ep*) and news crawl (*crawl*) test data. Feature groups are 12 dense features (default), rule identifiers (id), rule n-gram (ng), and rule shape (shape). Statistical significance at  $p$ -level  $< 0.05$  of a result difference on the test set to a different algorithm applied to the same feature group is indicated by raised algorithm number. <sup>†</sup> indicates statistically significant differences to best result across features groups for same algorithm, indicated in **bold face**. @ indicates the optimal number of epochs chosen on the devtest set.

and tuning the full set of 180,000 features are not significant. However, scaling all features to the full training set shows significant improvements for algorithm 3, and especially for algorithm 4, which gains 0.8 BLEU points over tuning 12 features on the development set. The number of features rises to 4.7 million without feature selection, which iteratively selects 100,000 features with best  $\ell_2$  norm values across shards. Feature templates such as rule n-grams and rule shapes only work if iterative mixing (algorithm 3) or feature selection (algorithm 4) are used. Adding rule id features works in combination with other sparse features.

Table 3 shows results for algorithms 1 and 4 on the Europarl data (*ep*) for different devtest and test sets. Europarl data were used in all runs for training and for setting the meta-parameter of number of epochs. Testing was done on the Europarl test set and news crawl test data from the years 2010 and 2011. Here tuning large feature sets on the respective dev sets yields significant improvements of around 2 BLEU points over tuning the 12 default features on the dev sets. Another 0.5 BLEU points (test-*crawl11*) or even 1.3 BLEU points (test-*crawl10*) are gained when scaling to the full training set using iterative features selection. Result differences on the Europarl test set were not significant for moving from dev to full train set. Algorithms 2 and 3 were infeasible to run on Europarl data beyond one epoch because features vectors grew too large to be kept in memory.

## 6 Discussion

We presented an approach to scaling discriminative learning for SMT not only to large feature

sets but also to large sets of parallel training data. Since inference for SMT (unlike many other learning problems) is very expensive, especially on large training sets, good parallelization is key. Our approach is made feasible and effective by applying joint feature selection across distributed stochastic learning processes. Furthermore, our local features are efficiently computable at runtime. Our algorithms and features are generic and can easily be re-implemented and make our results relevant across datasets and language pairs.

In future work, we would like to investigate more sophisticated features, better learners, and in general improve the components of our system that have been neglected in the current investigation of relative improvements by scaling the size of data and feature sets. Ultimately, since our algorithms are inspired by multi-task learning, we would like to apply them to scenarios where a natural definition of tasks is given. For example, patent data can be characterized along the dimensions of patent classes and patent text fields (Wäschle and Riezler, 2012) and thus are well suited for multi-task translation.

## Acknowledgments

Stefan Riezler and Patrick Simianer were supported in part by DFG grant “Cross-language Learning-to-Rank for Patent Retrieval”. Chris Dyer was supported in part by a MURI grant “The linguistic-core approach to structured translation and analysis of low-resource languages” from the US Army Research Office and a grant “Unsupervised Induction of Multi-Nonterminal Grammars for SMT” from Google, Inc.

## References

- Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable models for statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT'08)*, Columbus, OH.
- Léon Bottou. 2004. Stochastic learning. In Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch, editors, *Advanced Lectures on Machine Learning*, pages 146–168. Springer, Berlin.
- Olivier Chapelle and S. Sathya Keerthi. 2010. Efficient algorithms for ranking with SVMs. *Information Retrieval Journal*.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP'08)*, Waikiki, Honolulu, Hawaii.
- David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'09)*, Boulder, CO.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, Ann Arbor, MI.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2).
- Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the conference on Empirical Methods in Natural Language Processing (EMNLP'02)*, Philadelphia, PA.
- Ronan Collobert and Samy Bengio. 2004. Links between perceptrons, MLPs, and SVMs. In *Proceedings of the 21st International Conference on Machine Learning (ICML'04)*, Banff, Canada.
- Jeffrey Dean and Sanjay Ghemawat. 2004. Mapreduce: Simplified data processing on large clusters. In *Proceedings of the 6th Symposium on Operating System Design and Implementation (OSDI'04)*, San Francisco, CA.
- Kevin Duh and Katrin Kirchhoff. 2008. Beyond log-linear models: Boosted minimum error rate training for n-best ranking. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL'08), Short Paper Track*, Columbus, OH.
- Kevin Duh, Katsuhito Sudoh, Hajime Tsukada, Hideki Isozaki, and Masaaki Nagata. 2010. N-best reranking by multitask learning. In *Proceedings of the 5th Joint Workshop on Statistical Machine Translation and MetricsMATR*, Uppsala, Sweden.
- Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, Uppsala, Sweden.
- Chris Dyer, Kevin Gimpel, Jonathan H. Clark, and Noah A. Smith. 2011. The CMU-ARK german-english translation system. In *Proceedings of the 6th Workshop on Machine Translation (WMT11)*, Edinburgh, UK.
- Chris Dyer. 2009. Using a maximum entropy model to build segmentation lattices for MT. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT'09)*, Boulder, CO.
- Yoav Freund and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Journal of Machine Learning Research*, 37:277–296.
- Kevin Gimpel and Noah A. Smith. 2012. Structured ramp loss minimization for machine translation. In *Proceedings of 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2012)*, Montreal, Canada.
- Katsuhiko Hayashi, Taro Watanabe, Hajime Tsukada, and Hideki Isozaki. 2009. Structural support vector machines for log-linear approach in statistical machine translation. In *Proceedings of IWSLT*, Tokyo, Japan.
- Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation (WMT'11)*, Edinburgh, UK.
- Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP'11)*, Edinburgh, Scotland.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*, Edmonton, Canada.
- Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum Bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the 47th Annual Meeting of the Association for Computational*

- Linguistics and the 4th IJCNLP of the AFNLP (ACL-IJCNLP'09)*, Suntec, Singapore.
- Thomas Navin Lal, Olivier Chapelle, Jason Weston, and André Elisseeff. 2006. Embedded methods. In I.M. Guyon, S.R. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction: Foundations and Applications*. Springer.
- Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006a. An end-to-end discriminative approach to machine translation. In *Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL'06)*, Sydney, Australia.
- Percy Liang, Ben Taskar, and Dan Klein. 2006b. Alignment by agreement. In *Proceedings of the Human Language Technology Conference - North American Chapter of the Association for Computational Linguistics annual meeting (HLT-NAACL'06)*, New York, NY.
- Adam Lopez. 2007. Hierarchical phrase-based translation with suffix arrays. In *Proceedings of EMNLP-CoNLL*, Prague, Czech Republic.
- David McAllester and Joseph Keshet. 2011. Generalization bounds and consistency for latent structural probit and ramp loss. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS 2011)*, Granada, Spain.
- Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'10)*, Los Angeles, CA.
- Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley, New York.
- Guillaume Obozinski, Ben Taskar, and Michael I. Jordan. 2010. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20:231–252.
- Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL'00)*, Hongkong, China.
- Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, PA.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*, Edmonton, Canada.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Technical Report IBM Research Division Technical Report, RC22176 (W0190-022), Yorktown Heights, N.Y.
- Simon Perkins, Kevin Lacker, and James Theiler. 2003. Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research*, 3:1333–1356.
- Stefan Riezler and John Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL-05 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, Ann Arbor, MI.
- Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. 2007. Pegasos: Primal Estimated sub-Gradient Solver for SVM. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, Corvallis, OR.
- Libin Shen and Aravind K. Joshi. 2005. Ranking and reranking with perceptron. *Journal of Machine Learning Research*, 60(1-3):73–96.
- Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *Proceedings of the Human Language Technology conference / North American chapter of the Association for Computational Linguistics annual meeting (HLT/NAACL'04)*, Boston, MA.
- Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, Denver, CO.
- Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical MT. In *Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL'06)*, Sydney, Australia.
- Katharina Wäschele and Stefan Riezler. 2012. Structural and topical dimensions in multi-task patent translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, Avignon, France.
- Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2006. NTT statistical machine translation for IWSLT 2006. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, Kyoto, Japan.
- Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007*



*Joint Conference on Empirical Methods in Natural Language Processing and Computational Language Learning (EMNLP'07)*, Prague, Czech Republic.

- Benjamin Wellington, Joseph Turian, and Dan Melamed. 2009. Toward purely discriminative training for tree-structured translation models. In Cyril Goutte, Nicola Cancedda, and Marc Dymetman, editors, *Learning Machine Translation*, pages 132–149, Cambridge, MA. The MIT Press.
- Martin A. Zinkevich, Markus Weimer, Alex Smola, and Lihong Li. 2010. Parallelized stochastic gradient descent. In *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS'10)*, Vancouver, Canada.
- Andreas Zollmann and Khalil Sima'an. 2005. A consistent and efficient estimator for data-oriented parsing. *Journal of Automata, Languages and Combinatorics*, 10(2/3):367–388.

# Prediction of Learning Curves in Machine Translation

Prasanth Kolachina\* Nicola Cancedda† Marc Dymetman† Sriram Venkatapathy†

\* LTRC, IIIT-Hyderabad, Hyderabad, India

† Xerox Research Centre Europe, 6 chemin de Maupertuis, 38240 Meylan, France

## Abstract

Parallel data in the domain of interest is the key resource when training a statistical machine translation (SMT) system for a specific purpose. Since ad-hoc manual translation can represent a significant investment in time and money, a prior assesment of the amount of training data required to achieve a satisfactory accuracy level can be very useful. In this work, we show how to *predict* what the learning curve would look like *if* we were to manually translate increasing amounts of data.

We consider two scenarios, 1) Monolingual samples in the source and target languages are available and 2) An additional small amount of parallel corpus is also available. We propose methods for predicting learning curves in both these scenarios.

## 1 Introduction

Parallel data in the domain of interest is the key resource when training a statistical machine translation (SMT) system for a specific business purpose. In many cases it is possible to allocate some budget for manually translating a limited sample of relevant documents, be it via professional translation services or through increasingly fashionable crowdsourcing. However, it is often difficult to predict how much training data will be required to achieve satisfactory translation accuracy, preventing sound provisional budgetting. This prediction, or more generally the prediction of the *learning curve* of an SMT system as a function of available in-domain parallel data, is the objective of this paper.

We consider two scenarios, representative of realistic situations.

1. In the first scenario (S1), the SMT developer is given only monolingual source and target samples from the relevant domain, and a small *test* parallel corpus.

---

\*This research was carried out during an internship at Xerox Research Centre Europe.

2. In the second scenario (S2), an additional small *seed* parallel corpus is given that can be used to train small in-domain models and measure (with some variance) the evaluation score at a few points on the initial portion of the learning curve.

In both cases, the task consists in predicting an evaluation score (BLEU, throughout this work) on the test corpus as a function of the size of a subset of the source sample, assuming that we could have it manually translated and use the resulting bilingual corpus for training.

In this paper we provide the following contributions:

1. An extensive study across six parametric function families, empirically establishing that a certain three-parameter power-law family is well suited for modeling learning curves for the Moses SMT system when the evaluation score is BLEU. Our methodology can be easily generalized to other systems and evaluation scores (Section 3);
2. A method for *inferring* learning curves based on features computed from the resources available in scenario S1, suitable for both the scenarios described above (S1) and (S2) (Section 4);
3. A method for *extrapolating* the learning curve from a few measurements, suitable for scenario S2 (Section 5);
4. A method for *combining* the two approaches above, achieving on S2 better prediction accuracy than either of the two in isolation (Section 6).

In this study we limit tuning to the mixing parameters of the Moses log-linear model through MERT, keeping all meta-parameters (e.g. maximum phrase length, maximum allowed distortion, etc.) at their default values. One can expect further tweaking to lead to performance improvements, but this was a

necessary simplification in order to execute the tests on a sufficiently large scale.

Our experiments involve 30 distinct language pair and domain combinations and 96 different learning curves. They show that without any parallel data we can predict the expected translation accuracy at 75K segments within an error of 6 BLEU points (Table 4), while using a seed training corpus of 10K segments narrows this error to within 1.5 points (Table 6).

## 2 Related Work

Learning curves are routinely used to illustrate how the performance of experimental methods depend on the amount of training data used. In the SMT area, Koehn et al. (2003) used learning curves to compare performance for various meta-parameter settings such as *maximum phrase length*, while Turchi et al. (2008) extensively studied the behaviour of learning curves under a number of test conditions on Spanish-English. In Birch et al. (2008), the authors examined corpus features that contribute most to the machine translation performance. Their results showed that the most predictive features were the morphological complexity of the languages, their linguistic relatedness and their word-order divergence; in our work, we make use of these features, among others, for predicting translation accuracy (Section 4).

In a Machine Learning context, Perlich et al. (2003) used learning curves for predicting *maximum performance* bounds of learning algorithms and to compare them. In Gu et al. (2001), the learning curves of two classification algorithms were modelled for eight different large data sets. This work uses similar *a priori* knowledge for restricting the form of learning curves as ours (see Section 3), and also similar empirical evaluation criteria for comparing curve families with one another. While both application and performance metric in our work are different, we arrive at a similar conclusion that a power law family of the form  $y = c - a x^{-\alpha}$  is a good model of the learning curves.

Learning curves are also frequently used for determining empirically the number of iterations for an incremental learning procedure.

The crucial difference in our work is that in the previous cases, learning curves are plotted *a posteriori* i.e. once the labelled data has become available and the training has been performed, whereas

in our work the learning curve itself is the object of the prediction. Our goal is to learn to *predict* what the learning curve will be *a priori* without having to label the data at all (S1), or through labelling only a very small amount of it (S2).

In this respect, the academic field of Computational Learning Theory has a similar goal, since it strives to identify bounds to performance measures<sup>1</sup>, typically including a dependency on the training sample size. We take a purely empirical approach in this work, and obtain useful estimations for a case like SMT, where the complexity of the mapping between the input and the output prevents tight theoretical analysis.

## 3 Selecting a parametric family of curves

The first step in our approach consists in selecting a suitable family of shapes for the learning curves that we want to produce in the two scenarios being considered.

We formulate the problem as follows. For a certain bilingual test dataset  $d$ , we consider a set of observations  $O_d = \{(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)\}$ , where  $y_i$  is the performance on  $d$  (measured using BLEU (Papineni et al., 2002)) of a translation model trained on a parallel corpus of size  $x_i$ . The corpus size  $x_i$  is measured in terms of the number of segments (sentences) present in the parallel corpus.

We consider such observations to be generated by a regression model of the form:

$$y_i = F(x_i; \theta) + \epsilon_i \quad 1 \leq i \leq n \quad (1)$$

where  $F$  is a function depending on a vector parameter  $\theta$  which depends on  $d$ , and  $\epsilon_i$  is Gaussian noise of constant variance.

Based on our prior knowledge of the problem, we limit the search for a suitable  $F$  to families that satisfies the following conditions- monotonically increasing, concave and bounded. The first condition just says that more training data is better. The second condition expresses a notion of “diminishing returns”, namely that a given amount of additional training data is more advantageous when added to a small rather than to a big amount of initial data. The last condition is related to our use of BLEU — which is bounded by 1 — as a performance measure; It should be noted that some growth patterns which are sometimes proposed, such as a logarithmic regime of the form  $y \simeq a + b \log x$ , are not

<sup>1</sup>More often to a *loss*, which is equivalent.

compatible with this constraint.

We consider six possible families of functions satisfying these conditions, which are listed in Table 1. Preliminary experiments indicated that curves from

Model	Formula
Exp <sub>3</sub>	$y = c - e^{-ax+b}$
Exp <sub>4</sub>	$y = c - e^{-ax^\alpha+b}$
ExpP <sub>3</sub>	$y = c - e^{(x-b)^\alpha}$
Pow <sub>3</sub>	$y = c - ax^{-\alpha}$
Pow <sub>4</sub>	$y = c - (-ax + b)^{-\alpha}$
ILog <sub>2</sub>	$y = c - (a/\log x)$

Table 1: Curve families.

the ‘‘Power’’ and ‘‘Exp’’ family with only two parameters underfitted, while those with five or more parameters led to overfitting and solution instability. We decided to only select families with three or four parameters.

**Curve fitting technique** Given a set of observations  $\{(x_1, y_1), (x_2, y_2) \dots (x_n, y_n)\}$  and a curve family  $F(x; \theta)$  from Table 1, we compute a best fit  $\hat{\theta}$  where:

$$\hat{\theta} = \arg \min_{\theta} \sum_{i=1}^n [y_i - F(x_i; \theta)]^2, \quad (2)$$

through use of the *Levenberg-Marquardt* method (Moré, 1978) for non-linear regression.

For selecting a learning curve family, and for all other experiments in this paper, we trained a large number of systems on multiple *configurations* of training sets and sample sizes, and tested each on multiple *test* sets; these are listed in Table 2. All experiments use Moses (Koehn et al., 2007).<sup>2</sup>

Domain	Source Language	Target Language	# Test sets
Europarl (Koehn, 2005)	Fr, De, Es En	En Fr, De, Es	4
KFTT (Neubig, 2011)	Jp, En	En, Jp	2
EMEA (Tiedemann, 2009)	Da, De	En	4
News (Callison-Burch et al., 2011)	Cz, En, Fr, De, Es	Cz, En, Fr, De, Es	3

Table 2: The translation systems used for the curve fitting experiments, comprising 30 language-pair and domain combinations for a total of 96 learning curves. Language codes: Cz=Czech, Da=Danish, En=English, De=German, Fr=French, Jp=Japanese, Es=Spanish

The goodness of fit for each of the families is eval-

<sup>2</sup>The settings used in training the systems are those described in <http://www.statmt.org/wmt11/baseline.html>

uated based on their ability to *i)* fit over the entire set of observations, *ii)* extrapolate to points beyond the observed portion of the curve and *iii)* generalize well over different datasets.

We use a recursive fitting procedure where the curve obtained from fitting the first  $i$  points is used to predict the observations at two points:  $x_{i+1}$ , i.e. the point to the immediate right of the currently observed  $x_i$  and  $x_n$ , i.e. the largest point that has been observed.

The following error measures quantify the goodness of fit of the curve families:

1. Average root mean-squared error (RMSE):

$$\frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} \left\{ \frac{1}{n} \sum_{i=1}^n [y_i - F(x_i; \hat{\theta})]^2 \right\}_{ct}^{1/2}$$

where  $S$  is the set of training datasets,  $T_c$  is the set of test datasets for training configuration  $c$ ,  $\hat{\theta}$  is as defined in Eq. 2,  $N$  is the total number of combinations of training configurations and test datasets, and  $i$  ranges on a grid of training subset sizes. The expressions  $n, x_i, y_i, \hat{\theta}$  are all local to the combination  $ct$ .

2. Average root mean squared residual at next point  $X = x_{i+1}$  (NPR):

$$\frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} \left\{ \frac{1}{n-k-1} \sum_{i=k}^{n-1} [y_{i+1} - F(x_{i+1}; \hat{\theta}^i)]^2 \right\}_{ct}^{1/2}$$

where  $\hat{\theta}^i$  is obtained using only observations up to  $x_i$  in Eq. 2 and where  $k$  is the number of parameters of the family.<sup>3</sup>

3. Average root mean squared residual at the last point  $X = x_n$  (LPR):

$$\frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} \left\{ \frac{1}{n-k-1} \sum_{i=k}^{n-1} [y_n - F(x_n; \hat{\theta}^i)]^2 \right\}_{ct}^{1/2}$$

**Curve fitting evaluation** The evaluation of the goodness of fit for the curve families is presented in Table 3. The average values of the root mean-squared error and the average residuals across all the learning curves used in our experiments are shown in this table. The values are on the same scale as the BLEU scores. Figure 1 shows the curve fits obtained

<sup>3</sup>We start the summation from  $i = k$ , because at least  $k$  points are required for computing  $\hat{\theta}^i$ .

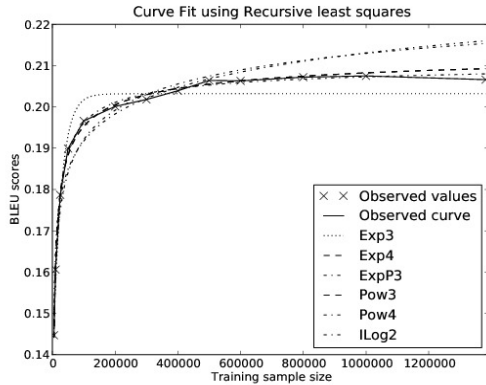


Figure 1: Curve fits using different curve families on a test dataset

for all the six families on a test dataset for English-German language pair.

Curve Family	RMSE	NPR	LPR
Exp <sub>3</sub>	0.0063	0.0094	0.0694
Exp <sub>4</sub>	0.0030	<i>0.0036</i>	<i>0.0072</i>
ExpP <sub>3</sub>	0.0040	0.0049	0.0145
<b>Pow<sub>3</sub></b>	<b>0.0029</b>	<b>0.0037</b>	<b>0.0091</b>
Pow <sub>4</sub>	<i>0.0026</i>	0.0042	0.0102
ILog <sub>2</sub>	0.0050	0.0067	0.0146

Table 3: Evaluation of the goodness of fit for the six families.

Looking at the values in Table 3, we decided to use the Pow<sub>3</sub> family as the best overall compromise. While it is not systematically better than Exp<sub>4</sub> and Pow<sub>4</sub>, it is good overall and has the advantage of requiring only 3 parameters.

#### 4 Inferring a learning curve from mostly monolingual data

In this section we address scenario S1: we have access to a source-language monolingual collection (from which portions to be manually translated could be sampled) and a target-language in-domain monolingual corpus, to supplement the target side of a parallel corpus while training a language model. The only available parallel resource is a very small test corpus. Our objective is to predict the evolution of the BLEU score on the given test set as a function of the size of a random subset of the training data

that we manually translate<sup>4</sup>. The intuition behind this is that the source-side and target-side monolingual data already convey significant information about the difficulty of the translation task.

We proceed in the following way. We first train models to predict the BLEU score at  $m$  anchor sizes  $s_1, \dots, s_m$ , based on a set of features globally characterizing the configuration of interest. We restrict our attention to linear models:

$$\mu_j = \mathbf{w}_j^\top \boldsymbol{\phi}, j \in \{1 \dots m\}$$

where  $\mathbf{w}_j$  is a vector of feature weights specific to predicting at anchor size  $j$ , and  $\boldsymbol{\phi}$  is a vector of size-independent configuration features, detailed below. We then perform inference using these models to predict the BLEU score at each anchor, for the test case of interest. We finally estimate the parameters of the learning curve by weighted least squares regression using the anchor predictions.

Anchor sizes can be chosen rather arbitrarily, but must satisfy the following two constraints:

1. They must be three or more in number in order to allow fitting the tri-parameter curve.
2. They should be spread as much as possible along the range of sample size.

For our experiments, we take  $m = 3$ , with anchors at 10K, 75K and 500K segments.

The feature vector  $\boldsymbol{\phi}$  consists of the following features:

1. General properties: number and average length of sentences in the (source) test set.
2. Average length of tokens in the (source) test set and in the monolingual source language corpus.
3. Lexical diversity features:
  - (a) type-token ratios for n-grams of order 1 to 5 in the monolingual corpus of both source and target languages
  - (b) perplexity of language models of order 2 to 5 derived from the monolingual source corpus computed on the source side of the test corpus.

<sup>4</sup>We specify that it is a random sample as opposed to a subset deliberately chosen to maximize learning effectiveness. While there are clear ties between our present work and active learning, we prefer to keep these two aspects distinct at this stage, and intend to explore this connection in future work.

4. Features capturing divergence between languages in the pair:
  - (a) average ratio of source/target sentence lengths in the test set.
  - (b) ratio of type-token ratios of orders 1 to 5 in the monolingual corpus of both source and target languages.
5. Word-order divergence: The divergence in the word-order between the source and the target languages can be captured using the part-of-speech (pos) tag sequences across languages. We use cross-entropy measure to capture similarity between the n-gram distributions of the pos tags in the monolingual corpora of the two languages. The order of the n-grams ranges between  $n = 2, 4 \dots 12$  in order to account for long distance reordering between languages. The pos tags for the languages are mapped to a reduced set of twelve pos tags (Petrov et al., 2012) in order to account for differences in tagsets used across languages.

These features capture our intuition that translation is going to be harder if the language in the domain is highly variable and if the source and target languages diverge more in terms of morphology and word-order.

The weights  $w_j$  are estimated from data. The training data for fitting these linear models is obtained in the following way. For each configuration (combination of language pair and domain)  $c$  and test set  $t$  in Table 2, a *gold curve* is fitted using the selected tri-parameter power-law family using a fine grid of corpus sizes. This is available as a byproduct of the experiments for comparing different parametric families described in Section 3. We then compute the value of the gold curves at the  $m$  anchor sizes: we thus have  $m$  “gold” vectors  $\mu_1, \dots, \mu_m$  with accurate estimates of BLEU at the anchor sizes<sup>5</sup>. We construct the design matrix  $\Phi$  with one column for each feature vector  $\phi_{ct}$  corresponding to each combination of training configuration  $c$  and test set  $t$ .

We then estimate weights  $w_j$  using Ridge regression ( $L^2$  regularization):

$$w_j = \arg \min_w \|\Phi^\top w - \mu_j\|^2 + C\|w\|^2 \quad (3)$$

<sup>5</sup>Computing these values from the gold curve rather than directly from the observations has the advantage of smoothing the observed values and also does not assume that observations at the anchor sizes are always directly available.

where the regularization parameter  $C$  is chosen by cross-validation. We also run experiments using Lasso ( $L^1$ ) regularization (Tibshirani, 1994) instead of Ridge. As baseline, we take a *constant mean* model predicting, for each anchor size  $s_j$ , the average of all the  $\mu_{jct}$ .

We do not assume the difficulty of predicting BLEU at all anchor points to be the same. To allow for this, we use (non-regularized) *weighted* least-squares to fit a curve from our parametric family through the  $m$  anchor points<sup>6</sup>. Following (Croarkin and Tobias, 2006, Section 4.4.5.2), the *anchor confidence* is set to be the inverse of the cross-validated mean square residuals:

$$\omega_j = \left( \frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} (\phi_{ct}^\top w_j^c - \mu_{jct})^2 \right)^{-1} \quad (4)$$

where  $w_j^c$  are the feature weights obtained by the regression above on all training configurations except  $c$ ,  $\mu_{jct}$  is the gold value at anchor  $j$  for training/test combination  $c, t$ , and  $N$  is the total number of such combinations<sup>7</sup>. In other words, we assign to each anchor point a confidence inverse to the cross-validated mean squared error of the model used to predict it.

For a new unseen configuration with feature vector  $\phi_u$ , we determine the parameters  $\theta_u$  of the corresponding learning curve as:

$$\theta_u = \arg \min_{\theta} \sum_j \omega_j (F(s_j; \theta) - \phi_u^\top w_j)^2 \quad (5)$$

## 5 Extrapolating a learning curve fitted on a small parallel corpus

Given a small “seed” parallel corpus, the translation system can be used to train small in-domain models and the evaluation score can be measured at a few initial sample sizes  $\{(x_1, y_1), (x_2, y_2) \dots (x_p, y_p)\}$ . The performance of the system for these initial points provides evidence for predicting its performance for larger sample sizes.

In order to do so, a learning curve from the family  $\text{Pow}_3$  is first fit through these initial points. We

<sup>6</sup>When the number of anchor points is the same as the number of parameters in the parametric family, the curve can be fit exactly through all anchor points. However the general discussion is relevant in case there are more anchor points than parameters, and also in view of the combination of inference and extrapolation in Section 6.

<sup>7</sup>Curves on different test data for the same training configuration are highly correlated and are therefore left out.

assume that  $p \geq 3$  for this operation to be well-defined. The best fit  $\hat{\eta}$  is computed using the same curve fitting as in Eq. 2.

At each individual anchor size  $s_j$ , the accuracy of prediction is measured using the root mean-squared error between the prediction of extrapolated curves and the gold values:

$$\left( \frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} [F(s_j; \hat{\eta}_{ct}) - \mu_{ctj}]^2 \right)^{1/2} \quad (6)$$

where  $\hat{\eta}_{ct}$  are the parameters of the curve fit using the initial points for the combination  $ct$ .

In general, we observed that the extrapolated curve tends to over-estimate BLEU for large samples.

## 6 Combining inference and extrapolation

In scenario S2, the models trained from the seed parallel corpus and the features used for inference (Section 4) provide complementary information. In this section we combine the two to see if this yields more accurate learning curves.

For the inference method of Section 4, predictions of models at anchor points are weighted by the inverse of the model empirical squared error ( $\omega_j$ ). We extend this approach to the extrapolated curves. Let  $u$  be a new configuration with seed parallel corpus of size  $x_u$ , and let  $x_l$  be the largest point in our grid for which  $x_l \leq x_u$ . We first train translation models and evaluate scores on samples of size  $x_1, \dots, x_l$ , fit parameters  $\hat{\eta}_u$  through the scores, and then extrapolate BLEU at the anchors  $s_j$ :  $F(s_j; \hat{\eta}_u)$ ,  $j \in \{1, \dots, m\}$ . Using the models trained for the experiments in Section 3, we estimate the squared extrapolation error at the anchors  $s_j$  when using models trained on size up to  $x_l$ , and set the confidence in the extrapolations<sup>8</sup> for  $u$  to its inverse:

$$\xi_j^{<l} = \left( \frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} (F(s_j; \eta_{ct}^{<l}) - \mu_{ctj})^2 \right)^{-1} \quad (7)$$

where  $N$ ,  $S$ ,  $T_c$  and  $\mu_{ctj}$  have the same meaning as in Eq. 4, and  $\eta_{ct}^{<l}$  are parameters fitted for configuration  $c$  and test  $t$  using only scores measured at  $x_1, \dots, x_l$ . We finally estimate the parameters  $\theta_u$  of

the combined curve as:

$$\theta_u = \arg \min_{\theta} \sum_j \omega_j (F(s_j; \theta) - \phi_u^\top \mathbf{w}_j)^2 + \xi_j^{<l} (F(s_j; \theta) - F(s_j; \hat{\eta}_u))^2$$

where  $\phi_u$  is the feature vector for  $u$ , and  $w_j$  are the weights we obtained from the regression in Eq. 3.

## 7 Experiments

In this section, we report the results of our experiments on predicting the learning curves.

### 7.1 Inferred Learning Curves

Regression model	10K	75K	500K
Ridge	0.063	<b>0.060</b>	<b>0.053</b>
Lasso	<b>0.054</b>	<b>0.060</b>	0.062
Baseline	0.112	0.121	0.121

Table 4: Root mean squared error of the linear regression models for each anchor size

In the case of inference from mostly monolingual data, the accuracy of the predictions at each of the anchor sizes is evaluated using root mean-squared error over the predictions obtained in a *leave-one-out* manner over the set of configurations from Table 2. Table 4 shows these results for Ridge and Lasso regression models at the three anchor sizes. As an example, the model estimated using Lasso for the 75K anchor size exhibits a root mean squared error of 6 BLEU points. The errors we obtain are lower than the error of the baseline consisting in taking, for each anchor size  $s_j$ , the average of all the  $\mu_{ctj}$ . The Lasso regression model selected four features from the entire feature set: *i*) Size of the test set (sentences & tokens) *ii*) Perplexity of language model (order 5) on the test set *iii*) Type-token ratio of the target monolingual corpus. Feature correlation measures such as Pearsons R showed that the features corresponding to type-token ratios of both source and target languages and size of test set have a high correlation with the BLEU scores at the three anchor sizes.

Figure 2 shows an instance of the inferred learning curves obtained using a weighted least squares method on the predictions at the anchor sizes. Table 7 presents the cumulative error of the inferred learning curves with respect to the gold curves, measured as the average distance between the curves in the range  $x \in [0.1K, 100K]$ .

<sup>8</sup>In some cases these can actually be interpolations.

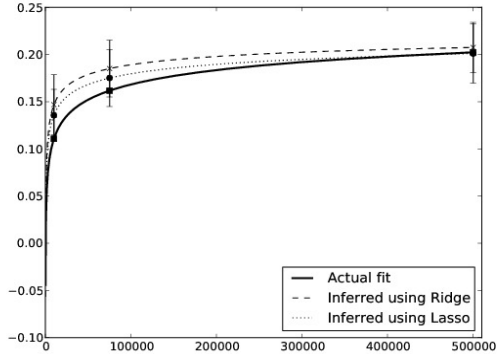


Figure 2: Inferred learning curve for English-Japanese test set. The error-bars show the *anchor confidence* for the predictions.

## 7.2 Extrapolated Learning Curves

As explained in Section 5, we evaluate the accuracy of predictions from the *extrapolated* curve using the root mean squared error (see Eq. 6) between the predictions of this curve and the gold values at the anchor points.

We conducted experiments for three sets of initial points, 1) 1K-5K-10K, 2) 5K-10K-20K, and 3) 1K-5K-10K-20K. For each of these sets, we show the prediction accuracy at the anchor sizes, 10K<sup>9</sup>, 75K, and 500K in Table 5.

Initial Points	10K	75K	500K
1K-5K-10K	0.005	0.017	0.042
5K-10K-20K	0.002	0.015	0.034
1K-5K-10K-20K	<b>0.002</b>	<b>0.008</b>	<b>0.019</b>

Table 5: Root mean squared error of the extrapolated curves at the three anchor sizes

The root mean squared errors obtained by extrapolating the learning curve are much lower than those obtained by prediction of translation accuracy using the monolingual corpus only (see Table 4), which is expected given that more direct evidence is available in the former case. In Table 5, one can also see that the root mean squared error for the sets 1K-5K-10K and 5K-10K-20K are quite close for anchor

<sup>9</sup>The 10K point is not an extrapolation point but lies within the range of the set of initial points. However, it does give a measure of the closeness of the curve fit using only the initial points with the gold fit using all the points; the value of this gold fit at 10K is not necessarily equal to the observation at 10K.

sizes 75K and 500K. However, when a configuration of four initial points is used for the same amount of “seed” parallel data, it outperforms both the configurations with three initial points.

## 7.3 Combined Learning Curves and Overall Comparison

In Section 6, we presented a method for combining the predicted learning curves from inference and extrapolation by using a weighted least squares approach. Table 6 reports the root mean squared error at the three anchor sizes from the combined curves.

Initial Points	Model	10K	75K	500K
1K-5K-10K	Ridge	0.005	0.015	0.038
	Lasso	0.005	0.014	0.038
5K-10K-20K	Ridge	0.001	0.006	0.018
	Lasso	0.001	0.006	0.018
1K-5K-10K-20K	Ridge	<b>0.001</b>	<b>0.005</b>	<b>0.014</b>
	Lasso	<b>0.001</b>	<b>0.005</b>	<b>0.014</b>

Table 6: Root mean squared error of the combined curves at the three anchor sizes

We also present an overall evaluation of all the predicted learning curves. The evaluation metric is the average distance between the predicted curves and the gold curves, within the range of sample sizes  $x_{min}=0.1K$  to  $x_{max}=500K$  segments; this metric is defined as:

$$\frac{1}{N} \sum_{c \in S} \sum_{t \in T_c} \frac{\sum_{x=x_{min}}^{x_{max}} |F(x; \hat{\eta}_{ct}) - F(x; \hat{\theta}_{ct})|}{x_{max} - x_{min}}$$

where  $\hat{\eta}_{ct}$  is the curve of interest,  $\hat{\theta}_{ct}$  is the gold curve, and  $x$  is in the range  $[x_{min}, x_{max}]$ , with a step size of 1. Table 7 presents the final evaluation.

Initial Points	IR	IL	EC	CR	CL
1K-5K-10K	0.034	0.050	0.018	0.015	<b>0.014</b>
5K-10K-20K	0.036	0.048	0.011	0.010	<b>0.009</b>
1K-5K-10K-20K	0.032	0.049	0.008	<b>0.007</b>	<b>0.007</b>

Table 7: Average distance of different predicted learning curves relative to the gold curve. Columns: IR=“Inference using Ridge model”, IL=“Inference using Lasso model”, EC=“Extrapolated curve”, CR=“Combined curve using Ridge”, CL=“Combined curve using Lasso”

We see that the combined curves (CR and CL) perform slightly better than the inferred curves (IR



and IL) and the extrapolated curves (EC). The average distance is on the same scale as the BLEU score, which suggests that our best curves can predict the gold curve within 1.5 BLEU points on average (the best result being 0.7 BLEU points when the initial points are 1K-5K-10K-20K) which is a telling result. The distances between the predicted and the gold curves for all the learning curves in our experiments are shown in Figure 3.

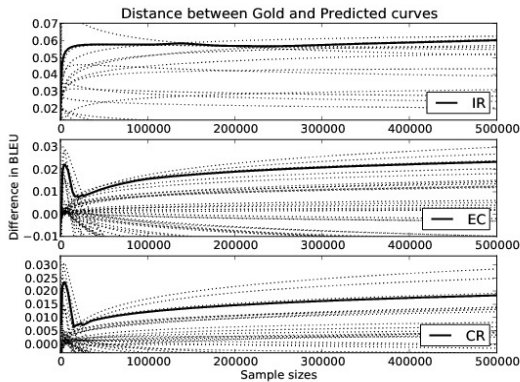


Figure 3: Distances between the predicted and the gold learning curves in our experiments across the range of sample sizes. The dotted lines indicate the distance from gold curve for each instance, while the bold line indicates the 95<sup>th</sup> quantile of the distance between the curves. IR=“Inference using Ridge model”, EC=“Extrapolated curve”, CR=“Combined curve using Ridge”.

We also provide a comparison of the different predicted curves with respect to the gold curve as shown in Figure 4.

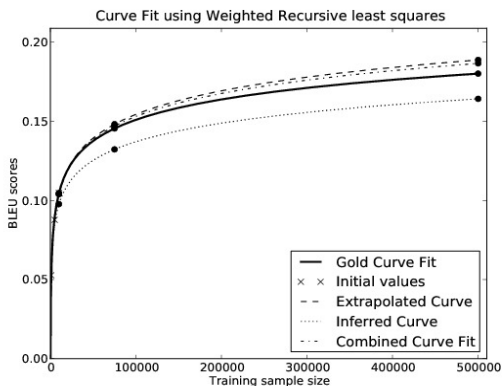


Figure 4: Predicted curves in the three scenarios for Czech-English test set using the Lasso model

## 8 Conclusion

The ability to predict the amount of parallel data required to achieve a given level of quality is very valuable in planning business deployments of statistical machine translation; yet, we are not aware of any rigorous proposal for addressing this need.

Here, we proposed methods that can be directly applied to predicting learning curves in realistic scenarios. We identified a suitable parametric family for modeling learning curves via an extensive empirical comparison. We described an *inference* method that requires a minimal initial investment in the form of only a small parallel *test* dataset. For the cases where a slightly larger in-domain “seed” parallel corpus is available, we introduced an *extrapolation* method and a *combined* method yielding high-precision predictions: using models trained on up to 20K sentence pairs we can predict performance on a given test set with a root mean squared error in the order of 1 BLEU point at 75K sentence pairs, and in the order of 2-4 BLEU points at 500K. Considering that variations in the order of 1 BLEU point on a same test dataset can be observed simply due to the instability of the standard MERT parameter tuning algorithm (Foster and Kuhn, 2009; Clark et al., 2011), we believe our results to be close to what can be achieved in principle. Note that by using gold curves as labels instead of actual measures we implicitly average across many rounds of MERT (14 for each curve), greatly attenuating the impact of the instability in the optimization procedure due to randomness.

For enabling this work we trained a multitude of instances of the same phrase-based SMT system on 30 distinct combinations of language-pair and domain, each with fourteen distinct training sets of increasing size and tested these instances on multiple in-domain datasets, generating 96 learning curves. BLEU measurements for all 96 learning curves along with the gold curves and feature values used for inferring the learning curves are available as additional material to this submission.

We believe that it should be possible to use insights from this paper in an active learning setting, to select, from an available monolingual source, a subset of a given size for manual translation, in such a way as to yield the highest performance, and we plan to extend our work in this direction.

## References

- Alexandra Birch, Miles Osborne, and Philipp Koehn. 2008. Predicting Success in Machine Translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 745–754, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 Workshop on Statistical Machine Translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 22–64, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 176–181, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Carroll Croarkin and Paul Tobias. 2006. *NIST/SEMATECH e-Handbook of Statistical Methods*. NIST/SEMATECH, July. Available online: <http://www.itl.nist.gov/div898/handbook/>.
- George Foster and Roland Kuhn. 2009. Stabilizing Minimum Error Rate Training. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 242–249, Athens, Greece, March. Association for Computational Linguistics.
- Baohua Gu, Feifang Hu, and Huan Liu. 2001. Modelling Classification Performance for Large Data Sets. In *Proceedings of the Second International Conference on Advances in Web-Age Information Management, WAIM '01*, pages 317–328, London, UK. Springer-Verlag.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of Human Language Technologies: The 2003 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 48–54, Edmonton, Canada, May. Association for Computational Linguistics.
- 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 the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.
- Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the 10th Machine Translation Summit*, Phuket, Thailand, September.
- Jorge J. Moré. 1978. The Levenberg-Marquardt Algorithm: Implementation and Theory. *Numerical Analysis. Proceedings Biennial Conference Dundee 1977*, 630:105–116.
- Graham Neubig. 2011. The Kyoto Free Translation Task. <http://www.phontron.com/kfft>.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.
- Claudia Perlich, Foster J. Provost, and Jeffrey S. Simonoff. 2003. Tree Induction vs. Logistic Regression: A Learning-Curve Analysis. *Journal of Machine Learning Research*, 4:211–255.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A Universal Part-of-Speech Tagset. In *Proceedings of the Eighth conference on International Language Resources and Evaluation (LREC'12)*, Istanbul, May. European Language Resources Association (ELRA).
- Robert Tibshirani. 1994. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.
- Jörg Tiedemann. 2009. News from OPUS - A Collection of Multilingual Parallel Corpora with Tools and Interfaces. In *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.
- Marco Turchi, Tijn De Bie, and Nello Cristianini. 2008. Learning Performance of a Machine Translation System: a Statistical and Computational Analysis. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 35–43, Columbus, Ohio, June. Association for Computational Linguistics.

# Probabilistic Integration of Partial Lexical Information for Noise Robust Haptic Voice Recognition

**Khe Chai Sim**

Department of Computer Science  
National University of Singapore  
13 Computing Drive, Singapore 117417  
simkc@comp.nus.edu.sg

## Abstract

This paper presents a probabilistic framework that combines multiple knowledge sources for Haptic Voice Recognition (HVR), a multimodal input method designed to provide efficient text entry on modern mobile devices. HVR extends the conventional voice input by allowing users to provide complementary partial lexical information via touch input to improve the efficiency and accuracy of voice recognition. This paper investigates the use of the initial letter of the words in the utterance as the partial lexical information. In addition to the acoustic and language models used in automatic speech recognition systems, HVR uses the haptic and partial lexical models as additional knowledge sources to reduce the recognition search space and suppress confusions. Experimental results show that both the word error rate and runtime factor can be reduced by a factor of two using HVR.

## 1 Introduction

Nowadays, modern portable devices, such as the smartphones and tablets, are equipped with microphone and touchscreen display. With these devices becoming increasingly popular, there is an urgent need for an efficient and reliable text entry method on these small devices. Currently, text entry using an onscreen virtual keyboard is the most widely adopted input method on these modern mobile devices. Unfortunately, typing with a small virtual keyboard can sometimes be cumbersome and frustratingly slow for many people. Instead of using

a virtual keyboard, it is also possible to use handwriting gestures to input text. Handwriting input offers a more convenient input method for writing systems with complex orthography, including many Asian languages such as Chinese, Japanese and Korean. However, handwriting input is not necessarily more efficient compared to keyboard input for English. Moreover, handwriting recognition is susceptible to recognition errors, too.

Voice input offers a hands-free solution for text entry. This is an attractive alternative for text entry because it completely eliminates the need for typing. Voice input is also more natural and faster for human to convey messages. Normally, the average human speaking rate is approximately 100 words per minute (WPM). Clarkson et al. (2005) showed that the typing speed for regular users reaches only 86.79 – 98.31 using a full-size keyboard and 58.61 – 61.44 WPM using a mini-QWERTY keyboard. Evidently, speech input is the preferred text entry method, provided that speech signals can be reliably and efficiently converted into texts. Unfortunately, voice input relies on automatic speech recognition (ASR) (Rabiner, 1989) technology, which requires high computational resources and is susceptible to performance degradation due to acoustic interference, such as the presence of noise.

In order to improve the reliability and efficiency of ASR, Haptic Voice Recognition (HVR) was proposed by Sim (2010) as a novel multimodal input method combining both speech and touch inputs. Touch inputs are used to generate *haptic events*, which correspond to the initial letters of the words in the spoken utterance. In addition to the regular beam

pruning used in traditional ASR (Ortmanns et al., 1997), search paths which are inconsistent with the haptic events are also pruned away to achieve further reduction in the recognition search space. As a result, the runtime of HVR is generally more efficient than ASR. Furthermore, haptic events are not susceptible to acoustic distortion, making HVR more robust to noise.

This paper proposes a probabilistic framework that encompasses multiple knowledge sources for combining the speech and touch inputs. This framework allows coherent probabilistic models of different knowledge sources to be tightly integrated. In addition to the acoustic model and language model used in ASR, *haptic model* and *partial lexical model* are also introduced to facilitate the integration of more sophisticated haptic events, such as the keystrokes, into HVR.

The remaining of this paper is organised as follows. Section 2 gives an overview of existing techniques in the literature that aim at improving noise robustness for automatic speech recognition. Section 3 gives a brief introduction to HVR. Section 4 proposes a probabilistic framework for HVR that unifies multiple knowledge sources as an integrated probabilistic generative model. Next, Section 5 describes how multiple knowledge sources can be integrated using Weighted Finite State Transducer (WFST) operations. Experimental results are presented in Section 6. Finally, conclusions are given in Section 7.

## 2 Noise Robust ASR

As previously mentioned, the process of converting speech into text using ASR is error-prone, where significant performance degradation is often due to the presence of noise or other acoustic interference. Therefore, it is crucial to improve the robustness of voice input in noisy environment. There are many techniques reported in the literature which aim at improving the robustness of ASR in noisy environment. These techniques can be largely divided into two groups: 1) using speech enhancement techniques to increase the signal-to-noise ratio of the noisy speech (Ortega-Garcia and Gonzalez-Rodriguez, 1996); and 2) using model-based compensation schemes to *adapt* the acoustic models to

noisy environment (Gales and Young, 1996; Acero et al., 2000).

From the information-theoretic point of view, in order to achieve reliable information transmission, *redundancies* are introduced so that information lost due to channel distortion or noise corruption can be recovered. Similar concept can also be applied to improve the robustness of voice input in noisy environment. Additional complementary information can be provided using other input modalities to provide *cues* (redundancies) to boost the recognition performance. The next section will introduce a multimodal interface that combines speech and touch inputs to improve the efficiency and noise robustness for text entry using a technique known as Haptic Voice Recognition (Sim, 2010).

## 3 Haptic Voice Recognition (HVR)

For many voice-enabled applications, users often find voice input to be a *black box* that captures the users' voice and automatically converts it into texts using ASR. It does not provide much flexibility for human intervention through other modalities in case of errors. Certain applications may return multiple hypotheses, from which users can choose the most appropriate output. Any remaining errors are typically corrected manually. However, it may be more useful to give users more control during the input stage, instead of having a post-processing step for error correction. This motivates the investigation of multimodal interface that tightly integrates speech input with other modalities.

Haptic Voice Recognition (HVR) is a multimodal interface designed to offer users the opportunity to add his or her '*magic touch*' in order to improve the accuracy, efficiency and robustness of voice input. HVR is designed for modern mobile devices equipped with an embedded microphone to capture speech signals and a touchscreen display to receive touch events. The HVR interface aims to combine both speech and touch modalities to enhance speech recognition. When using an HVR interface, users will input text verbally, at the same time provide additional cues in the form of *Partial Lexical Information* (PLI) to guide the recognition search. PLIs are simplified lexical representation of words that should be easy to enter whilst speaking (*e.g.* the

prefix and/or suffix letters). Preliminary simulated experiments conducted by Sim (2010) show that potential performance improvements both in terms of recognition speed and noise robustness can be achieved using the initial letters as PLIs. For example, to enter the text “*Henry will be in Boston next Friday*”, the user will speak the sentence and enter the following letter sequence: ‘H’, ‘W’, ‘B’, ‘I’, ‘B’, ‘N’ and ‘F’. These additional letter sequence is simple enough to be entered whilst speaking; and yet they provide crucial information that can significantly improve the efficiency and robustness of speech recognition. For instance, the number of letters entered can be used to constrain the number of words in the recognition output, thereby suppressing spurious insertion and deletion errors, which are commonly observed in noisy environment. Furthermore, the identity of the letters themselves can be used to guide the search process so that partial word sequences in the search graph that do not conform to the PLIs provided by the users can be pruned away.

PLI provides additional complementary information that can be used to eliminate confusions caused by poor speech signal. In conventional ASR, acoustically similar word sequences are typically resolved implicitly using a language model where contexts of neighboring words are used for disambiguation. On the other hand, PLI can also be very effective in disambiguating homophones<sup>1</sup> and similar sounding words and phrases that have distinct initial letters. For example, ‘*hour*’ versus ‘*our*’, ‘*vary*’ versus ‘*marry*’ and ‘*great wine*’ versus ‘*grey twine*’.

This paper considers two methods of generating the initial letter sequence using a touchscreen. The first method requires the user to tap on the appropriate keys on an onscreen virtual keyboard to generate the desired letter sequence. This method is similar to that proposed in Sim (2010). However, typing on small devices like smartphones may require a great deal of concentration and precision from the users. Alternatively, the initial letters can be entered using handwriting gestures. A gesture recognizer can be used to determine the letters entered by the users. In order to achieve high recognition accuracy, each letter is represented by a single-stroke gesture, so that isolated letter recognition can be performed. Fig-

<sup>1</sup>Words with the same pronunciation

ure 1 shows the single-stroke gestures that are used in this work.

#### 4 A Probabilistic Formulation for HVR

Let  $\mathcal{O} = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T\}$  denote a sequence of  $T$  observed acoustic features such as MFCC (Davis and Mermelstein, 1980) or PLP (Hermansky, 1990) and  $\mathcal{H} = \{\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_N\}$  denote a sequence of  $N$  haptic features. For the case of keyboard input, each  $\mathbf{h}_i$  is a discrete symbol representing one of the 26 letters. On the other hand, for handwriting input, each  $\mathbf{h}_i$  represents a sequence of 2-dimensional vectors that corresponds to the coordinates of the points of the keystroke. Therefore, the haptic voice recognition problem can be defined as finding the *joint* optimal solution for both the word sequence,  $\hat{\mathcal{W}}$  and the PLI sequence,  $\hat{\mathcal{L}}$ , given  $\mathcal{O}$  and  $\mathcal{H}$ . This can be expressed using the following formulation:

$$(\hat{\mathcal{W}}, \hat{\mathcal{L}}) = \arg \max_{\mathcal{W}, \mathcal{L}} P(\mathcal{W}, \mathcal{L} | \mathcal{O}, \mathcal{H}) \quad (1)$$

where according to the Bayes’ theorem:

$$\begin{aligned} P(\mathcal{W}, \mathcal{L} | \mathcal{O}, \mathcal{H}) &= \frac{p(\mathcal{O}, \mathcal{H} | \mathcal{W}, \mathcal{L}) P(\mathcal{W}, \mathcal{L})}{p(\mathcal{O}, \mathcal{H})} \\ &= \frac{p(\mathcal{O} | \mathcal{W}) p(\mathcal{H} | \mathcal{L}) P(\mathcal{W}, \mathcal{L})}{p(\mathcal{O}, \mathcal{H})} \end{aligned} \quad (2)$$

The joint prior probability of the observed inputs,  $p(\mathcal{O}, \mathcal{H})$ , can be discarded during the maximisation of Eq. 1 since it is independent of  $\mathcal{W}$  and  $\mathcal{L}$ .  $p(\mathcal{O} | \mathcal{W})$  is the acoustic likelihood of the word sequence,  $\mathcal{W}$ , generating the acoustic feature sequence,  $\mathcal{O}$ . Similarly,  $P(\mathcal{H} | \mathcal{L})$  is the *haptic likelihood* of the lexical sequence,  $\mathcal{L}$ , generating the observed haptic inputs,  $\mathcal{H}$ . The *joint* prior probability,  $P(\mathcal{W}, \mathcal{L})$ , can be decomposed into:

$$P(\mathcal{W}, \mathcal{L}) = P(\mathcal{L} | \mathcal{W}) P(\mathcal{W}) \quad (3)$$

where  $P(\mathcal{W})$  can be modelled by the word-based  $n$ -gram language model (Chen and Goodman, 1996) commonly used in automatic speech recognition. Combining Eq. 2 and Eq. 3 yields:

$$\begin{aligned} P(\mathcal{W}, \mathcal{L} | \mathcal{O}, \mathcal{H}) &\propto \\ &p(\mathcal{O} | \mathcal{W}) \times p(\mathcal{H} | \mathcal{L}) \times P(\mathcal{L} | \mathcal{W}) \times P(\mathcal{W}) \end{aligned} \quad (4)$$

It is evident from the above equation that the probabilistic formulation of HVR combines four knowledge sources:

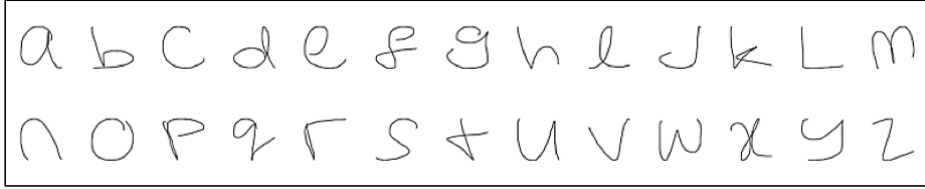


Figure 1: Examples of single-stroke handwriting gestures for the 26 English letters

- Acoustic model score:  $p(\mathcal{O}|\mathcal{W})$
- Haptic model score:  $p(\mathcal{H}|\mathcal{L})$
- PLI model score:  $P(\mathcal{L}|\mathcal{W})$
- Language model score:  $P(\mathcal{W})$

Note that the acoustic model and language model scores are already used in the conventional ASR. The probabilistic formulation of HVR incorporated two additional probabilities: *haptic model score*,  $p(\mathcal{H}|\mathcal{L})$  and *PLI model score*,  $P(\mathcal{L}|\mathcal{W})$ . The role of the haptic model and PLI model will be described in the following sub-sections.

#### 4.1 Haptic Model

Similar to having an acoustic model as a statistical representation of the phoneme sequence generating the observed acoustic features, a haptic model is used to model the PLI sequence generating the observed haptic inputs,  $\mathcal{H}$ . The haptic likelihood can be factorised as

$$p(\mathcal{H}|\mathcal{L}) = \prod_{i=1}^N p(\mathbf{h}_i|l_i) \quad (5)$$

where  $\mathcal{L} = \{l_i : 1 \leq i \leq N\}$ .  $l_i$  is the  $i$ th PLI in  $\mathcal{L}$  and  $\mathbf{h}_i$  is the  $i$ th haptic input feature. In this work, each PLI represent the initial letter of a word. Therefore,  $l_i$  represents one of the 26 letters. As previously mentioned, for keyboard input,  $\mathbf{h}_i$  are discrete features whose values are also one of the 26 letters. Therefore,  $p(\mathbf{h}_i|l_i)$  forms a  $26 \times 26$  matrix. A simple model can be derived by making  $p(\mathbf{h}_i|l_i)$  an identity matrix. Therefore,  $p(\mathbf{h}_i|l_i) = 1$  if  $\mathbf{h}_i = l_i$ ; otherwise,  $p(\mathbf{h}_i|l_i) = 0$ . However, it is also possible to have a non-diagonal matrix for  $p(\mathbf{h}_i|l_i)$  in order to accommodate typing errors, so that non-zero probabilities are assigned to cases where  $\mathbf{h}_i \neq l_i$ .

For handwriting input,  $\mathbf{h}_i$  denote a sequence of 2-dimensional feature vectors, which can be modelled using Hidden Markov Models (HMMs) (Rabiner, 1989). Therefore,  $(\mathbf{h}_i|l_i)$  is simply given by the HMM likelihood. In this work, each of the 26 letters is represented by a left-to-right HMM with 3 emitting states.

#### 4.2 Partial Lexical Information (PLI) Model

Finally, a PLI model is used to impose the compatibility constraint between the PLI sequence,  $\mathcal{L}$ , and the word sequence,  $\mathcal{W}$ . Let  $\mathcal{W} = \{\mathbf{w}_i : 1 \leq i \leq M\}$  denote a word sequence of length  $M$ . If  $M = N$ , the PLI model likelihood,  $P(\mathcal{L}|\mathcal{W})$ , can be expressed in the following form:

$$P(\mathcal{L}|\mathcal{W}) = \prod_{i=1}^N P(l_i|\mathbf{w}_i) \quad (6)$$

where  $P(l_i|\mathbf{w}_i)$  is the likelihood of the  $i$ th word,  $\mathbf{w}_i$ , generating the  $i$ th PLI,  $l_i$ . Since each word is represented by a unique PLI (the initial letter) in this work, the PLI model score is given by

$$P(l_i|\mathbf{w}_i) = C_{\text{sub}} = \begin{cases} 1 & \text{if } l_i = \text{initial letter of } \mathbf{w}_i \\ 0 & \text{otherwise} \end{cases}$$

On the other hand, if  $N \neq M$ , *insertions* and *deletions* have to be taken into consideration:

$$P(l_i = \epsilon|\mathbf{w}_i) = C_{\text{del}} \quad \text{and} \quad P(l_i|\mathbf{w}_i = \epsilon) = C_{\text{ins}}$$

where  $\epsilon$  represents an empty token.  $C_{\text{del}}$  and  $C_{\text{ins}}$  denote the deletion and insertion penalties respectively. This work assumes  $C_{\text{del}} = C_{\text{ins}} = 0$ . This means that the word count of the HVR output matches the length of the initial letter sequence entered by the user. Assigning a non-zero value to  $C_{\text{del}}$  gives the users option to skip entering letters for certain words (*e.g.* short words).

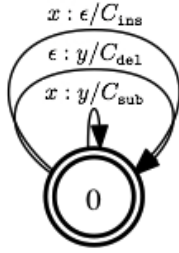


Figure 2: WSFT representation of PLI model,  $\bar{\mathcal{P}}$

## 5 Integration of Knowledge Sources

As previously mentioned, the HVR recognition process involves maximising the posterior probability in Eq. 4, which can be expressed in terms of four knowledge sources. It turns out that these knowledge sources can be represented as Weighted Finite State Transducers (WFSTs) (Mohri et al., 2002) and the *composition* operation ( $\circ$ ) can be used to integrate these knowledge sources into a single WFST:

$$\bar{\mathcal{F}}_{\text{integrated}} = \bar{\mathcal{A}} \circ \bar{\mathcal{L}} \circ \bar{\mathcal{P}} \circ \bar{\mathcal{H}} \quad (7)$$

where  $\bar{\mathcal{A}}$ ,  $\bar{\mathcal{L}}$ ,  $\bar{\mathcal{P}}$  and  $\bar{\mathcal{H}}$  denote the WFST representation of the acoustic model, language model, PLI model and haptic model respectively. Mohri et al. (2002) has shown that Hidden Markov Models (HMMs) and  $n$ -gram language models can be viewed as WFSTs. Furthermore, HMM-based haptic models are also used in this work to represent the single-stroke letters shown in Fig. 1. Therefore,  $\bar{\mathcal{A}}$ ,  $\bar{\mathcal{L}}$ , and  $\bar{\mathcal{H}}$  can be obtained from the respective probabilistic models. Finally, the PLI model described in Section 4.2 can also be represented using the WFST as shown in Fig. 2. The transition weights of these WFSTs are given by the negative log probability of the respective models.  $\bar{\mathcal{P}}$  can be viewed as a merger that defines the possible alignments between the speech and haptic inputs. Each complete path in  $\bar{\mathcal{F}}$  represents a valid pair of  $\mathcal{W}$  and  $\mathcal{L}$  such that the weight of the path is given by the *negative log*  $P(\mathcal{L}, \mathcal{W} | \mathcal{O}, \mathcal{H})$ . Therefore, finding the shortest path in  $\bar{\mathcal{F}}$  is equivalent to solving Eq. 1.

Direct decoding from the overall composed WFST,  $\bar{\mathcal{F}}_{\text{integrated}}$ , is referred to as *integrated decoding*. Alternatively, HVR can also operate in a *lattice rescoring* manner. Speech input and haptic input are processed separately by the ASR system and the haptic model respectively. The ASR system may

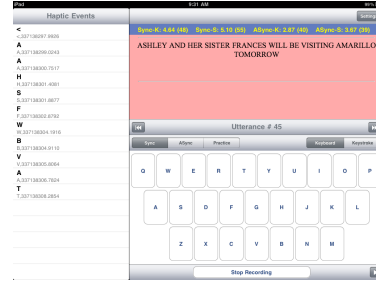


Figure 3: Screenshot depicting the HVR prototype operating with keyboard input

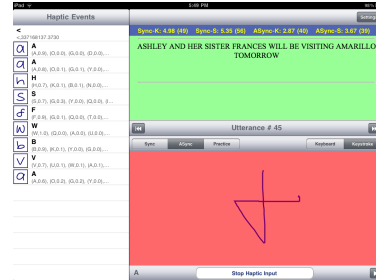


Figure 4: Screenshot depicting the HVR prototype operating with keystroke input

generate multiple hypotheses of word sequences in the form of a lattice. Similarly, the haptic model may also generate a lattice containing the most probably letter sequences. Let  $\hat{\mathcal{L}}$  and  $\hat{\mathcal{H}}$  represent the word and letter lattices respectively. Then, the final HVR output can be obtained by searching for the shortest path of the following merged WFST:

$$\bar{\mathcal{F}}_{\text{rescore}} = \hat{\mathcal{L}} \circ \bar{\mathcal{P}} \circ \hat{\mathcal{H}} \quad (8)$$

Note that the above composition may yield an empty WFST. This may happen if the lattices generated by the ASR system or the haptic model are not large enough to produce any valid pair of  $\mathcal{W}$  and  $\mathcal{L}$ .

## 6 Experimental Results

In this section, experimental results are reported based on the data collected using a prototype HVR interface implemented on an iPad. This prototype HVR interface allows both speech and haptic input data to be captured either synchronously or asynchronously and the partial lexical information can be entered using either a soft keyboard or handwriting gestures. Figures 3 and 4 shows the screenshot of the HVR prototype iPad app using the key-

Donna was in Cincinnati last Thursday.
Adam will be visiting Charlotte tomorrow
Janice will be in Chattanooga next month.
Christine will be visiting Corpus Christi next Tuesday.

Table 1: Example sentences used for data collection.

board and keystroke inputs respectively. Therefore, there are altogether four input configurations. For each configuration, 250 sentences were collected from a non-native fluent English speaker. 200 sentences were used as test data while the remaining 50 sentences were used for acoustic model adaptation. These sentences contain a variety of given names, surnames and city names so that confusions cannot be easily resolved using a language model. Example sentences used for data collection are shown in Table 1. In order to investigate the robustness of HVR in noisy environment, the collected speech data were also artificially corrupted with additive babble noise from the NOISEX database (Varga and Steeneken, 1993) to synthesise noisy speech signal-to-noise (SNR) levels of 20 and 10 decibels<sup>2</sup>.

The ASR system used in all the experiments reported in this paper consists of a set of HMM-based triphone acoustic models and an  $n$ -gram language model. The HMM models were trained using 39-dimensional MFCC features. Each HMM has a left-to-right topology and three emitting states. The emission probability for each state is represented by a single Gaussian component<sup>3</sup>. A bigram language model with a vocabulary size of 200 words was used for testing. The acoustic models were also noise-compensated using VTS (Acero et al., 2000) in order to achieve a better baseline performance.

### 6.1 Comparison of Input Speed

Table 2 shows the speech, letter and total input speed using different input configurations. For synchronous HVR, the total input speed is the same as the speech and letter input speed since both the speech and haptic inputs are provided concurrently. According to this study, synchronous keyboard input speed is 86 words per minutes (WPM). This is

<sup>2</sup>Higher SNR indicates a better speech quality

<sup>3</sup>A single Gaussian component system was used as a compromise between speed and accuracy for mobile apps.

Haptic Input	HVR Mode	Input Speed (WPM)		
		Speech	Letter	Total
Keyboard	Sync	86	86	86
	ASync	100	105	51
Keystroke	Sync	78	78	78
	ASync	97	83	45

Table 2: Comparison of the speech and letter input speed, measured in Words-Per-Minute (WPM), for different HVR input configurations

slightly faster than keystroke input using handwriting gestures, where the input speed is 78 WPM. This is not surprising since key taps are much quicker to generate compared to handwriting gestures. On the other hand, the individual speech and letter input speed are faster for asynchronous mode because users do not need to multi-task. However, since the speech and haptic inputs are provided concurrently, the resulting total input speed for asynchronous HVR is much slower compared to synchronous HVR. Therefore, synchronous HVR is potentially more efficient than asynchronous HVR.

### 6.2 Performance of ASR

HVR Mode	SNR	WER (%)	LER (%)
ASync	Clean	22.2	17.0
	20 dB	30.2	24.2
	10 dB	33.3	28.5
Sync (Keyboard)	Clean	25.9	20.2
	20 dB	34.6	28.8
	10 dB	35.5	29.9
Sync (Keystroke)	Clean	29.0	22.5
	20 dB	40.1	32.0
	10 dB	37.9	31.3

Table 3: WER and LER performance of ASR in different noise conditions

First of all, the Word Error Rate (WER) and Letter Error Rate (LER) performances for standard ASR systems in different noise conditions are summarized in Table 3. These are results using pure ASR, without adding the haptic inputs. Speech recorded using asynchronous HVR is considered normal speech. The ASR system achieved 22.2%, 30.2% and 33.3% WER in clean, 20dB and 10dB



conditions respectively. Note that the acoustic models have been compensated using VTS (Acero et al., 2000) for noisy conditions. Table 3 also shows the system performance considering on the initial letter sequence of the recognition output. This indicates the potential improvements that can be obtained with the additional first letter information. Note that the pure ASR system output contains substantial initial letter errors.

For synchronous HVR, the recorded speech is expected to exhibit different characteristics since it may be influenced by concurrent haptic input. Table 3 shows that there are performance degradations, both in terms of WER and LER, when performing ASR on these speech utterances. Also, the degradations caused by simultaneous keystroke input are greater. The degradation may be caused by phenomena such as the presence of *filled pauses* and the *lengthening* of phoneme duration. Other forms of disfluencies may have also been introduced to the realized speech utterances. Nevertheless, the additional information provided by the PLIs will outweigh these degradations.

### 6.3 Performance of Synchronous HVR

Haptic Input	SNR	WER (%)	LER (%)
Keyboard	Clean	11.8	1.1
	20 dB	12.7	1.0
	10 dB	15.0	1.0
Keystroke	Clean	11.4	0.3
	20 dB	13.1	0.9
	10 dB	14.0	1.0

Table 4: WER and LER performance of synchronous HVR in different noise conditions

The performance of synchronous HVR is shown in Table 4. Compared to the results shown in Table 3, the WER performance of synchronous HVR improved by approximately a factor of two. Furthermore, the LER performance improved significantly. For keyboard input, the LER reduced to about 1.0% for all noise conditions. Note that the tradeoffs between the WER and LER performance can be adjusted by applying appropriately weights to different knowledge sources during integration. For keystroke input, top five letter candidates returned

by the handwriting recognizer were used. Therefore, in clean condition, the acoustic models are able to recover some of the errors introduced by the handwriting recognizer, bringing the LER down to as low as 0.3%. However, in noisy conditions, the LER performance is similar to those using keyboard input. Overall, synchronous and asynchronous HVR achieved WER comparable performance.

### 6.4 Performance of Asynchronous HVR

Haptic Input	SNR	WER (%)	LER (%)
Keyboard	Clean	10.2	0.6
	20 dB	11.2	0.6
	10 dB	13.0	0.6
Keystroke	Clean	10.7	0.4
	20 dB	11.4	1.0
	10 dB	13.4	1.1

Table 5: WER and LER performance of asynchronous HVR in different noise conditions

Similar to synchronous HVR, asynchronous HVR also achieved significant performance improvements over the pure ASR systems. Table 5 shows the WER and LER performance of asynchronous HVR in different noise conditions. The WER performance of asynchronous HVR is consistently better than that of synchronous HVR (comparing Tables 4 and 5). This is expected since the speech quality for asynchronous HVR is higher. However, considering the much slower input speed (*c.f.* Table 2) and the marginal WER improvements for asynchronous HVR, synchronous HVR appears to be a better configuration.

### 6.5 Integrated Decoding vs. Lattice Rescoring

SNR	WER (%)		
	Clean	20dB	10dB
Integrated	11.8	12.7	15.0
Lat-rescore	11.2	18.6	18.1

Table 6: WER performance of keyboard synchronous HVR using integrated decoding and lattice rescoring

As previously mentioned in Section 5, HVR can also be performed in two stages using lattice rescoring technique. Table 6 shows the performance

comparison between integrated decoding and lattice rescoring for HVR. Both methods gave similar performance in clean condition. However, lattice rescoring yielded significantly worse performance in noisy environment. Therefore, it is important to tightly integrate the PLI into the decoding process to avoid premature pruning away optimal paths.

## 6.6 Runtime Performance

ASR system searches for the best word sequence using a dynamic programming paradigm (Ney and Ortmanns, 1999). The complexity of the search increases with the vocabulary size as well as the length of the input speech. A well-known concept of *Token Passing* (Young et al., 1989) can be used to describe the recognition search process. A set of active tokens are being propagated upon observing an acoustic feature frame. The best token that survived to the end of the utterance represents the best output. Typically, beam pruning technique (Ortmanns et al., 1997) is applied to improve the recognition efficiency. Tokens which are unlikely to yield the optimal solution will be pruned away. HVR performs a more stringent pruning, where paths that do not conform to the PLI sequence are also be pruned away.

System	SNR	RT	Active Tokens Per Frame
ASR	Clean	1.9	6260
	20 dB	2.0	6450
	10 dB	2.4	7168
Keyboard	Clean	0.9	3490
	20 dB	0.9	3764
	10 dB	1.0	4442
Keystroke	Clean	1.1	4059
	20 dB	1.2	4190
	10 dB	1.5	4969

Table 7: WER and LER performance of integrated and rescoring synchronous HVR in different noise conditions

Table 7 shows the comparison of the runtime factors and the average number of active tokens per frame for ASR and HVR systems. The standard ASR system runs at 1.9, 2.0 and 2.4 times real-time (xRT)<sup>4</sup>. The runtime factor increases with de-

<sup>4</sup>Runtime factor is computed as the ratio between the recognition duration and the input speech duration

creasing SNR because the presence of noise introduces more confusions, which renders beam pruning (Ortmanns et al., 1997) less effective. The number of active tokens per frame also increases from 6260 to 7168 as the SNR drops from the clean condition to 10dB. On the other hand, there are significant speedup in the runtime of HVR systems. In particular, synchronous HVR achieved the best runtime performance, which is roughly consistent across different noise conditions (approximately 1.0 xRT). The average number of active tokens also reduces to the range of 3490 – 4442. Therefore, the synchronous HVR using keyboard input is robust to noisy environment, both in terms of WER and runtime performance. The runtime performance using keystroke input is also comparable to that using keyboard input (only slightly worse). Therefore, both keyboard and keystroke inputs are effective ways for entering the initial letters for HVR. However, it is worth noting that the iPad was used for the studies conducted in this work. The size of the iPad screen is sufficiently large to allow efficient keyboard entry. However, for devices with smaller screen, keystroke inputs may be easier to use and less error-prone.

## 7 Conclusions

This paper has presented a unifying probabilistic framework for the multimodal Haptic Voice Recognition (HVR) interface. HVR offers users the option to interact with the system using touchscreen during voice input so that additional *cues* can be provided to improve the efficiency and robustness of voice recognition. Partial Lexical Information (PLI), such as the initial letter of the words, are used as cues to guide the recognition search process. Therefore, apart from the acoustic and language models used in conventional ASR, HVR also combines the haptic model as well as the PLI model to yield an integrated probabilistic model. This probabilistic framework integrates multiple knowledge sources using the weighted finite state transducer operation. Such integration is achieved using the composition operation which can be applied on-the-fly to yield efficient implementation. Experimental results show that this framework can be used to achieve a more efficient and robust multimodal interface for text entry on modern portable devices.

## References

- Alex Acero, Li Deng, Trausti Kristjansson, and Jerry Zhang. 2000. HMM adaptation using vector Taylor series for noisy speech recognition. In *Proc. of ICSLP*, volume 3, pages 869–872.
- Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics, ACL '96*, pages 310–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Edward Clarkson, James Clawson, Kent Lyons, and Thad Starner. 2005. An empirical study of typing rates on mini-qwerty keyboards. In *CHI '05 extended abstracts on Human factors in computing systems, CHI EA '05*, pages 1288–1291, New York, NY, USA. ACM.
- S. B. Davis and P. Mermelstein. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustic, Speech and Signal Processing*, 28(4):357–366.
- Mark Gales and Steve Young. 1996. Robust continuous speech recognition using parallel model combination. *IEEE Transactions on Speech and Audio Processing*, 4:352–359.
- H. Hermansky. 1990. Perceptual Linear Predictive (PLP) analysis of speech. *Journal of the Acoustic Society of America*, 87(4):1738–1752.
- Mehryar Mohri, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. *Computer Speech & Language*, 16(1):69–88.
- H. Ney and S. Ortmanns. 1999. Dynamic programming search for continuous speech recognition. *IEEE Signal Processing Magazine*, 16(5):64–83.
- J. Ortega-Garcia and J. Gonzalez-Rodriguez. 1996. Overview of speech enhancement techniques for automatic speaker recognition. In *Proceedings of International Conference on Spoken Language (ICSLP)*, pages 929–932.
- S. Ortmanns, H. Ney, H. Coenen, and Eiden A. 1997. Look-ahead techniques for fast beam search. In *ICASSP*.
- L. A. Rabiner. 1989. A tutorial on hidden Markov models and selective applications in speech recognition. In *Proc. of the IEEE*, volume 77, pages 257–286, February.
- K. C. Sim. 2010. Haptic voice recognition: Augmenting speech modality with touch events for efficient speech recognition. In *Proc. SLT Workshop*.
- A. Varga and H.J.M. Steeneken. 1993. Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication*, 12(3):247–251.
- S.J. Young, N.H. Russell, and J.H.S Thornton. 1989. Token passing: a simple conceptual model for connected speech recognition systems. Technical report.

# A Nonparametric Bayesian Approach to Acoustic Model Discovery

Chia-ying Lee and James Glass

Computer Science and Artificial Intelligence Laboratory

Massachusetts Institute of Technology

Cambridge, MA 02139, USA

{chiaying, jrg}@csail.mit.edu

## Abstract

We investigate the problem of acoustic modeling in which prior language-specific knowledge and transcribed data are unavailable. We present an unsupervised model that simultaneously segments the speech, discovers a proper set of sub-word units (e.g., phones) and learns a Hidden Markov Model (HMM) for each induced acoustic unit. Our approach is formulated as a Dirichlet process mixture model in which each mixture is an HMM that represents a sub-word unit. We apply our model to the TIMIT corpus, and the results demonstrate that our model discovers sub-word units that are highly correlated with English phones and also produces better segmentation than the state-of-the-art unsupervised baseline. We test the quality of the learned acoustic models on a spoken term detection task. Compared to the baselines, our model improves the relative precision of top hits by at least 22.1% and outperforms a language-mismatched acoustic model.

## 1 Introduction

Acoustic models are an indispensable component of speech recognizers. However, the standard process of training acoustic models is expensive, and requires not only language-specific knowledge, e.g., the phone set of the language, a pronunciation dictionary, but also a large amount of transcribed data. Unfortunately, these necessary data are only available for a very small number of languages in the world. Therefore, a procedure for training acoustic models without annotated data would not only be a breakthrough from the traditional approach, but

would also allow us to build speech recognizers for any language efficiently.

In this paper, we investigate the problem of unsupervised acoustic modeling with only spoken utterances as training data. As suggested in Garcia and Gish (2006), unsupervised acoustic modeling can be broken down to three sub-tasks: segmentation, clustering segments, and modeling the sound pattern of each cluster. In previous work, the three sub-problems were often approached sequentially and independently in which initial steps are not related to later ones (Lee et al., 1988; Garcia and Gish, 2006; Chan and Lee, 2011). For example, the speech data was usually segmented regardless of the clustering results and the learned acoustic models.

In contrast to the previous methods, we approach the problem by modeling the three sub-problems as well as the unknown set of sub-word units as latent variables in one nonparametric Bayesian model. More specifically, we formulate a Dirichlet process mixture model where each mixture is a Hidden Markov Model (HMM) used to model a sub-word unit and to generate observed segments of that unit. Our model seeks the set of sub-word units, segmentation, clustering and HMMs that best represent the observed data through an iterative inference process. We implement the inference process using Gibbs sampling.

We test the effectiveness of our model on the TIMIT database (Garofolo et al., 1993). Our model shows its ability to discover sub-word units that are highly correlated with standard English phones and to capture acoustic context information. For the segmentation task, our model outperforms the state-of-

the-art unsupervised method and improves the relative F-score by 18.8 points (Dusan and Rabiner, 2006). Finally, we test the quality of the learned acoustic models through a keyword spotting task. Compared to the state-of-the-art unsupervised methods (Zhang and Glass, 2009; Zhang et al., 2012), our model yields a relative improvement in precision of top hits by at least 22.1% with only some degradation in equal error rate (EER), and outperforms a language-mismatched acoustic model trained with supervised data.

## 2 Related Work

**Unsupervised Sub-word Modeling** We follow the general guideline used in (Lee et al., 1988; Garcia and Gish, 2006; Chan and Lee, 2011) and approach the problem of unsupervised acoustic modeling by solving three sub-problems of the task: segmentation, clustering and modeling each cluster. The key difference, however, is that our model does not assume independence among the three aspects of the problem, which allows our model to refine its solution to one sub-problem by exploiting what it has learned about other parts of the problem. Second, unlike (Lee et al., 1988; Garcia and Gish, 2006) in which the number of sub-word units to be learned is assumed to be known, our model learns the proper size from the training data directly.

Instead of segmenting utterances, the authors of (Varadarajan et al., 2008) trained a single state HMM using all data at first, and then iteratively split the HMM states based on objective functions. This method achieved high performance in a phone recognition task using a label-to-phone transducer trained from some transcriptions. However, the performance seemed to rely on the quality of the transducer. For our work, we assume no transcriptions are available and measure the quality of the learned acoustic units via a spoken query detection task as in Jansen and Church (2011).

Jansen and Church (2011) approached the task of unsupervised acoustic modeling by first discovering repetitive patterns in the data, and then learned a whole-word HMM for each found pattern, where the state number of each HMM depends on the average length of the pattern. The states of the whole-word HMMs were then collapsed and used to represent

acoustic units. Instead of discovering repetitive patterns first, our model is able to learn from any given data.

**Unsupervised Speech Segmentation** One goal of our model is to segment speech data into small sub-word (e.g., phone) segments. Most unsupervised speech segmentation methods rely on acoustic change for hypothesizing phone boundaries (Scharenborg et al., 2010; Qiao et al., 2008; Dusan and Rabiner, 2006; Estevan et al., 2007). Even though the overall approaches differ, these algorithms are all one-stage and bottom-up segmentation methods (Scharenborg et al., 2010). Our model does not make a single one-stage decision; instead, it infers the segmentation through an iterative process and exploits the learned sub-word models to guide its hypotheses on phone boundaries.

**Bayesian Model for Segmentation** Our model is inspired by previous applications of nonparametric Bayesian models to segmentation problems in NLP and speaker diarization (Goldwater, 2009; Fox et al., 2011); particularly, we adapt the inference method used in (Goldwater, 2009) to our segmentation task. Our problem is, in principle, similar to the word segmentation problem discussed in (Goldwater, 2009). The main difference, however, is that our model is under the continuous real value domain, and the problem of (Goldwater, 2009) is under the discrete symbolic domain. For the domain our problem is applied to, our model has to include more latent variables and is more complex.

## 3 Problem Formulation

The goal of our model, given a set of spoken utterances, is to jointly learn the following:

- Segmentation: To find the phonetic boundaries within each utterance.
- Nonparametric clustering: To find a proper set of clusters and group acoustically similar segments into the same cluster.
- Sub-word modeling: To learn a HMM to model each sub-word acoustic unit.

We model the three sub-tasks as latent variables in our approach. In this section, we describe the observed data, latent variables, and auxiliary variables

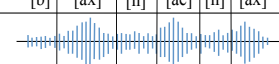
Pronunciation	b		a		n		a		n		a	
	[b]	[ax]	[ax]	[ax]	[n]	[ae]	[n]	[ax]	[n]	[ax]	[n]	[ax]
												
Frame index ( $t$ )	1	2	3	4	5	6	7	8	9	10	11	
Speech feature ( $x_t^i$ )	$x_1^i$	$x_2^i, x_3^i, x_4^i$	$x_5^i, x_6^i$	$x_7^i, x_8^i$	$x_9^i$	$x_{10}^i, x_{11}^i$						
Boundary variable ( $b_t^i$ )	1	0	0	1	0	1	0	1	1	0	1	
Boundary index ( $g_q^i$ )	$g_0^i$	$g_1^i$	$g_2^i$	$g_3^i$	$g_4^i$	$g_5^i$	$g_6^i$					
Segment ( $p_{j,k}^i$ )	$p_{1,1}^i$	$p_{2,4}^i$	$p_{5,6}^i$	$p_{7,8}^i$	$p_{9,9}^i$	$p_{10,11}^i$						
Duration ( $d_{j,k}^i$ )	1	3	2	2	1	2						
Cluster label ( $c_{j,k}^i$ )	$c_{1,1}^i$	$c_{2,4}^i$	$c_{5,6}^i$	$c_{7,8}^i$	$c_{9,9}^i$	$c_{10,11}^i$						
HMM ( $\theta_c$ )	$\theta_1$	$\theta_2$	$\theta_3$	$\theta_4$	$\theta_5$	$\theta_6$						
Hidden state ( $s_t^i$ )	1	1	2	3	1	3	1	3	1	1	3	
Mixture ID	1	1	6	8	3	7	5	2	8	2	8	

Figure 1: An example of the observed data and hidden variables of the problem for the word *banana*. See Section 3 for a detailed explanation.

of the problem and show an example in Fig. 1. In the next section, we show the generative process our model uses to generate the observed data.

**Speech Feature ( $x_t^i$ )** The only observed data for our problem are a set of spoken utterances, which are converted to a series of 25 ms 13-dimensional Mel-Frequency Cepstral Coefficients (MFCCs) (Davis and Mermelstein, 1980) and their first- and second-order time derivatives at a 10 ms analysis rate. We use  $x_t^i \in \mathbb{R}^{39}$  to denote the  $t^{\text{th}}$  feature frame of the  $i^{\text{th}}$  utterance. Fig. 1 illustrates how the speech signal of a single word utterance *banana* is converted to a sequence of feature vectors  $x_1^i$  to  $x_{11}^i$ .

**Boundary ( $b_t^i$ )** We use a binary variable  $b_t^i$  to indicate whether a phone boundary exists between  $x_t^i$  and  $x_{t+1}^i$ . If our model hypothesizes  $x_t^i$  to be the last frame of a sub-word unit, which is called a *boundary frame* in this paper,  $b_t^i$  is assigned with value 1; or 0 otherwise. Fig. 1 shows an example of the boundary variables where the values correspond to the true answers. We use an auxiliary variable  $g_q^i$  to denote the index of the  $q^{\text{th}}$  boundary frame in utterance  $i$ . To make the derivation of posterior distributions easier in Section 5, we define  $g_0^i$  to be the beginning of an utterance, and  $L_i$  to be the number of boundary frames in an utterance. For the example shown in Fig. 1,  $L_i$  is equal to 6.

**Segment ( $p_{j,k}^i$ )** We define a segment to be composed of feature vectors between two boundary frames. We use  $p_{j,k}^i$  to denote a segment that consists of  $x_j^i, x_{j+1}^i \dots x_k^i$  and  $d_{j,k}^i$  to denote the length of  $p_{j,k}^i$ . See Fig. 1 for more examples.

**Cluster Label ( $c_{j,k}^i$ )** We use  $c_{j,k}^i$  to specify the cluster label of  $p_{j,k}^i$ . We assume segment  $p_{j,k}^i$  is generated by the sub-word HMM with label  $c_{j,k}^i$ .

**HMM ( $\theta_c$ )** In our model, each HMM has three emission states, which correspond to the beginning, middle and end of a sub-word unit (Jelinek, 1976). A traversal of each HMM must start from the first state, and only left-to-right transitions are allowed even though we allow skipping of the middle and the last state for segments shorter than three frames. The emission probability of each state is modeled by a diagonal Gaussian Mixture Model (GMM) with 8 mixtures. We use  $\theta_c$  to represent the set of parameters that define the  $c^{\text{th}}$  HMM, which includes state transition probability  $a_c^{j,k}$ , and the GMM parameters of each state emission probability. We use  $w_{c,s}^m \in \mathbb{R}$ ,  $\mu_{c,s}^m \in \mathbb{R}^{39}$  and  $\lambda_{c,s}^m \in \mathbb{R}^{39}$  to denote the weight, mean vector and the diagonal of the inverse covariance matrix of the  $m^{\text{th}}$  mixture in the GMM for the  $s^{\text{th}}$  state in the  $c^{\text{th}}$  HMM.

**Hidden State ( $s_t^i$ )** Since we assume the observed data are generated by HMMs, each feature vector,  $x_t^i$ , has an associated hidden state index. We denote the hidden state of  $x_t^i$  as  $s_t^i$ .

**Mixture ID ( $m_t^i$ )** Similarly, each feature vector is assumed to be emitted by the state GMM it belongs to. We use  $m_t^i$  to identify the Gaussian mixture that generates  $x_t^i$ .

## 4 Model

We aim to discover and model a set of sub-word units that represent the spoken data. If we think of utterances as sequences of repeated sub-word units, then in order to find the sub-words, we need a model that concentrates probability on highly frequent patterns while still preserving probability for previously unseen ones. Dirichlet processes are particularly suitable for our goal. Therefore, we construct our model as a Dirichlet Process (DP) mixture model, of which the components are HMMs that are used

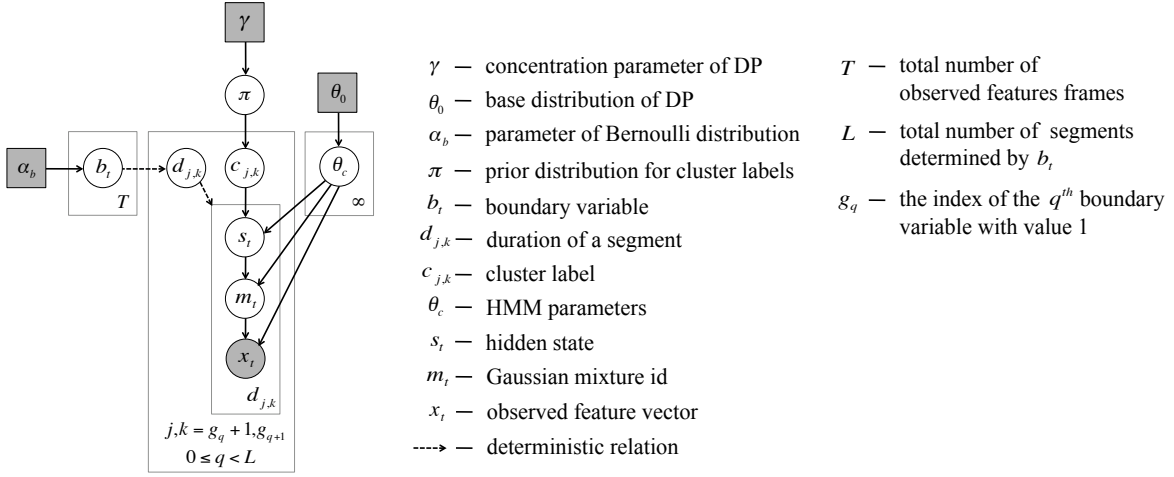


Figure 2: The graphical model for our approach. The shaded circle denotes the observed feature vectors, and the squares denote the hyperparameters of the priors used in our model. The dotted arrows indicate deterministic relations. Note that the Markov chain structure over the  $s_t$  variables is not shown here due to limited space.

to model sub-word units. We assume each spoken segment is generated by one of the clusters in this DP mixture model. Here, we describe the generative process our model uses to generate the observed utterances and present the corresponding graphical model. For clarity, we assume that the values of the boundary variables  $b_t^i$  are given in the generative process. In the next section, we explain how to infer their values.

Let  $p_{g_q^i+1, g_{q+1}^i}$  for  $0 \leq q \leq L_i - 1$  be the segments of the  $i^{th}$  utterance. Our model assumes each segment is generated as follows:

1. Choose a cluster label  $c_{g_q^i+1, g_{q+1}^i}^i$  for  $p_{g_q^i+1, g_{q+1}^i}^i$ . This cluster label can be either an existing label or a new one. Note that the cluster label determines which HMM is used to generate the segment.
2. Given the cluster label, choose a hidden state for each feature vector  $x_t^i$  in the segment.
3. For each  $x_t^i$ , based on its hidden state, choose a mixture from the GMM of the chosen state.
4. Use the chosen Gaussian mixture to generate the observed feature vector  $x_t^i$ .

The generative process indicates that our model ignores utterance boundaries and views the entire data as concatenated spoken segments. Given this

viewpoint, we discard the utterance index,  $i$ , of all variables in the rest of the paper.

The graphical model representing this generative process is shown in Fig. 2, where the shaded circle denotes the observed feature vectors, and the squares denote the hyperparameters of the priors used in our model. Specifically, we use a Bernoulli distribution as the prior of the boundary variables and impose a Dirichlet process prior on the cluster labels and the HMM parameters. The dotted arrows represent deterministic relations. For example, the boundary variables deterministically construct the duration of each segment,  $d$ , which in turn sets the number of feature vectors that should be generated for a segment. In the next section, we show how to infer the value of each of the latent variables in Fig. 2<sup>1</sup>.

## 5 Inference

We employ Gibbs sampling (Gelman et al., 2004) to approximate the posterior distribution of the hidden variables in our model. To apply Gibbs sampling to our problem, we need to derive the conditional posterior distributions of each hidden variable of the model. In the following sections, we first derive the sampling equations for each hidden variable and then describe how we incorporate acoustic cues to reduce the sampling load at the end.

<sup>1</sup>Note that the value of  $\pi$  is irrelevant to our problem; therefore, it is integrated out in the inference process

## 5.1 Sampling Equations

Here we present the sampling equations for each hidden variable defined in Section 3. We use  $P(\cdot|\dots)$  to denote a conditional posterior probability given observed data, all the other variables, and hyperparameters for the model.

**Cluster Label ( $c_{j,k}$ )** Let  $C$  be the set of distinctive label values in  $c_{-j,k}$ , which represents all the cluster labels except  $c_{j,k}$ . The conditional posterior probability of  $c_{j,k}$  for  $c \in C$  is:

$$P(c_{j,k} = c|\dots) \propto P(c_{j,k} = c|c_{-j,k}; \gamma)P(p_{j,k}|\theta_c) \\ = \frac{n^{(c)}}{N - 1 + \gamma}P(p_{j,k}|\theta_c) \quad (1)$$

where  $\gamma$  is a parameter of the DP prior. The first line of Eq. 1 follows Bayes' rule. The first term is the conditional prior, which is a result of the DP prior imposed on the cluster labels<sup>2</sup>. The second term is the conditional likelihood, which reflects how likely the segment  $p_{j,k}$  is generated by  $\text{HMM}_c$ . We use  $n^{(c)}$  to represent the number of cluster labels in  $c_{-j,k}$  taking the value  $c$  and  $N$  to represent the total number of segments in current segmentation.

In addition to existing cluster labels,  $c_{j,k}$  can also take a new cluster label, which corresponds to a new sub-word unit. The corresponding conditional posterior probability is:

$$P(c_{j,k} \neq c, c \in C|\dots) \propto \frac{\gamma}{N - 1 + \gamma} \int_{\theta} P(p_{j,k}|\theta) d\theta \quad (2)$$

To deal with the integral in Eq. 2, we follow the suggestions in (Rasmussen, 2000; Neal, 2000). We sample an HMM from the prior and compute the likelihood of the segment given the new HMM to approximate the integral.

Finally, by normalizing Eq. 1 and Eq. 2, the Gibbs sampler can draw a new value for  $c_{j,k}$  by sampling from the normalized distribution.

**Hidden State ( $s_t$ )** To enforce the assumption that a traversal of an HMM must start from the first state and end at the last state<sup>3</sup>, we do not sample hidden state indices for the first and the last frame of a segment. For each of the remaining feature vectors in

<sup>2</sup>See (Neal, 2000) for an overview on Dirichlet process mixture models and the inference methods.

<sup>3</sup>If a segment has only 1 frame, we assign the first state to it.

a segment  $p_{j,k}$ , we sample a hidden state index according to the conditional posterior probability:

$$P(s_t = s|\dots) \propto \\ P(s_t = s|s_{t-1})P(x_t|\theta_{c_{j,k}}, s_t = s)P(s_{t+1}|s_t = s) \\ = a_{c_{j,k}}^{s_{t-1}, s}P(x_t|\theta_{c_{j,k}}, s_t = s)a_{c_{j,k}}^{s, s_{t+1}} \quad (3)$$

where the first term and the third term are the conditional prior – the transition probability of the HMM that  $p_{j,k}$  belongs to. The second term is the likelihood of  $x_t$  being emitted by state  $s$  of  $\text{HMM}_{c_{j,k}}$ . Note for initialization,  $s_t$  is sampled from the first prior term in Eq. 3.

**Mixture ID ( $m_t$ )** For each feature vector in a segment, given the cluster label  $c_{j,k}$  and the hidden state index  $s_t$ , the derivation of the conditional posterior probability of its mixture ID is straightforward:

$$P(m_t = m|\dots) \\ \propto P(m_t = m|\theta_{c_{j,k}}, s_t)P(x_t|\theta_{c_{j,k}}, s_t, m_t = m) \\ = w_{c_{j,k}, s_t}^m P(x_t|\mu_{c_{j,k}, s_t}^m, \lambda_{c_{j,k}, s_t}^m) \quad (4)$$

where  $1 \leq m \leq 8$ . The conditional posterior consists of two terms: 1) the mixing weight of the  $m^{\text{th}}$  Gaussian in the state GMM indexed by  $c_{j,k}$  and  $s_t$  and 2) the likelihood of  $x_t$  given the Gaussian mixture. The sampler draws a value for  $m_t$  from the normalized distribution of Eq. 4.

**HMM Parameters ( $\theta_c$ )** Each  $\theta_c$  consists of two sets of variables that define an HMM: the state emission probabilities  $w_{c,s}^m, \mu_{c,s}^m, \lambda_{c,s}^m$  and the state transition probabilities  $a_c^{j,k}$ . In the following, we derive the conditional posteriors of these variables.

**Mixture Weight  $w_{c,s}^m$ :** We use  $\underline{w}_{c,s} = \{w_{c,s}^m | 1 \leq m \leq 8\}$  to denote the mixing weights of the Gaussian mixtures of state  $s$  of HMM  $c$ . We choose a symmetric Dirichlet distribution with a positive hyperparameter  $\beta$  as its prior. The conditional posterior probability of  $\underline{w}_{c,s}$  is:

$$P(\underline{w}_{c,s}|\dots) \propto P(\underline{w}_{c,s}; \beta)P(\mathbf{m}_{c,s}|\underline{w}_{c,s}) \\ \propto \text{Dir}(\underline{w}_{c,s}; \beta)\text{Mul}(\mathbf{m}_{c,s}; \underline{w}_{c,s}) \\ \propto \text{Dir}(\underline{w}_{c,s}; \beta') \quad (5)$$

where  $\mathbf{m}_{c,s}$  is the set of mixture IDs of feature vectors that belong to state  $s$  of HMM  $c$ . The  $m^{\text{th}}$  entry of  $\beta'$  is  $\beta + \sum_{m_t \in \mathbf{m}_{c,s}} \delta(m_t, m)$ , where we use  $\delta(\cdot)$



$$\begin{aligned}
P(p_{l,t}, p_{t+1,r} | \mathbf{c}^-, \boldsymbol{\theta}) &= P(p_{l,t} | \mathbf{c}^-, \boldsymbol{\theta}) P(p_{t+1,r} | \mathbf{c}^-, c_{l,t}, \boldsymbol{\theta}) \\
&= \left[ \sum_{c \in \mathcal{C}} \frac{n^{(c)}}{N^- + \gamma} P(p_{l,t} | \theta_c) + \frac{\gamma}{N^- + \gamma} \int_{\theta} P(p_{l,t} | \theta) d\theta \right] \\
&\quad \times \left[ \sum_{c \in \mathcal{C}} \frac{n^{(c)} + \delta(c_{l,t}, c)}{N^- + 1 + \gamma} P(p_{t+1,r} | \theta_c) + \frac{\gamma}{N^- + 1 + \gamma} \int_{\theta} P(p_{t+1,r} | \theta) d\theta \right] \\
P(p_{l,r} | \mathbf{c}^-, \boldsymbol{\theta}) &= \sum_{c \in \mathcal{C}} \frac{n^{(c)}}{N^- + \gamma} P(p_{l,r} | \theta_c) + \frac{\gamma}{N^- + \gamma} \int_{\theta} P(p_{l,r} | \theta) d\theta
\end{aligned}$$

Figure 3: The full derivation of the relative conditional posterior probabilities of a boundary variable.

to denote the discrete Kronecker delta. The last line of Eq. 5 comes from the fact that Dirichlet distributions are a conjugate prior for multinomial distributions. This property allows us to derive the update rule analytically.

**Gaussian Mixture**  $\mu_{c,s}^m, \lambda_{c,s}^m$ : We assume the dimensions in the feature space are independent. This assumption allows us to derive the conditional posterior probability for a single-dimensional Gaussian and generalize the results to other dimensions.

Let the  $d^{\text{th}}$  entry of  $\mu_{c,s}^m$  and  $\lambda_{c,s}^m$  be  $\mu_{c,s}^{m,d}$  and  $\lambda_{c,s}^{m,d}$ . The conjugate prior we use for the two variables is a normal-Gamma distribution with hyperparameters  $\mu_0, \kappa_0, \alpha_0$  and  $\beta_0$  (Murphy, 2007).

$$\begin{aligned}
&P(\mu_{c,s}^{m,d}, \lambda_{c,s}^{m,d} | \mu_0, \kappa_0, \alpha_0, \beta_0) \\
&= N(\mu_{c,s}^{m,d} | \mu_0, (\kappa_0 \lambda_{c,s}^{m,d})^{-1}) Ga(\lambda_{c,s}^{m,d} | \alpha_0, \beta_0)
\end{aligned}$$

By tracking the  $d^{\text{th}}$  dimension of feature vectors  $x \in \{x_t | m_t = m, s_t = s, c_{j,k} = c, x_t \in p_{j,k}\}$ , we can derive the conditional posterior distribution of  $\mu_{c,s}^{m,d}$  and  $\lambda_{c,s}^{m,d}$  analytically following the procedures shown in (Murphy, 2007). Due to limited space, we encourage interested readers to find more details in (Murphy, 2007).

**Transition Probabilities**  $a_c^{j,k}$ : We represent the transition probabilities at state  $j$  in HMM  $c$  using  $\underline{a}_c^j$ . If we view  $\underline{a}_c^j$  as mixing weights for states reachable from state  $j$ , we can simply apply the update rule derived for the mixing weights of Gaussian mixtures shown in Eq. 5 to  $\underline{a}_c^j$ . Assume we use a symmetric Dirichlet distribution with a positive hyperparameter  $\eta$  as the prior, the conditional posterior for  $\underline{a}_c^j$  is:

$$P(\underline{a}_c^j | \dots) \propto Dir(\underline{a}_c^j; \eta')$$

where the  $k^{\text{th}}$  entry of  $\eta'$  is  $\eta + n_c^{j,k}$ , the number of occurrences of the state transition pair  $(j, k)$  in segments that belong to HMM  $c$ .

**Boundary Variable** ( $b_t$ ) To derive the conditional posterior probability for  $b_t$ , we introduce two variables:

$$\begin{aligned}
l &= (\arg \max_{g_q} g_q < t) + 1 \\
r &= \arg \min_{g_q} t < g_q
\end{aligned}$$

where  $l$  is the index of the closest turned-on boundary variable that precedes  $b_t$  plus 1, while  $r$  is the index of the closest turned-on boundary variable that follows  $b_t$ . Note that because  $g_0$  and  $g_L$  are defined,  $l$  and  $r$  always exist for any  $b_t$ .

Note that the value of  $b_t$  only affects segmentation between  $x_l$  and  $x_r$ . If  $b_t$  is turned on, the sampler hypothesizes two segments  $p_{l,t}$  and  $p_{t+1,r}$  between  $x_l$  and  $x_r$ . Otherwise, only one segment  $p_{l,r}$  is hypothesized. Since the segmentation on the rest of the data remains the same no matter what value  $b_t$  takes, the conditional posterior probability of  $b_t$  is:

$$P(b_t = 1 | \dots) \propto P(p_{l,t}, p_{t+1,r} | \mathbf{c}^-, \boldsymbol{\theta}) \quad (6)$$

$$P(b_t = 0 | \dots) \propto P(p_{l,r} | \mathbf{c}^-, \boldsymbol{\theta}) \quad (7)$$

where we assume that the prior probabilities for  $b_t = 1$  and  $b_t = 0$  are equal;  $\mathbf{c}^-$  is the set of cluster labels of all segments except those between  $x_l$  and  $x_r$ ; and  $\boldsymbol{\theta}$  indicates the set of HMMs that have associated segments. Our Gibbs sampler hypothesizes  $b_t$ 's value by sampling from the normalized distribution of Eq. 6 and Eq. 7. The full derivations of Eq. 6 and Eq. 7 are shown in Fig. 3.

Note that in Fig. 3,  $N^-$  is the total number of segments in the data except those between  $x_l$  and  $x_r$ .

For  $b_t = 1$ , to account the fact that when the model generates  $p_{t+1,r}$ ,  $p_{l,t}$  is already generated and owns a cluster label, we sample a cluster label for  $p_{l,t}$  that is reflected in the Kronecker delta function. To handle the integral in Fig. 3, we sample one HMM from the prior and compute the likelihood using the new HMM to approximate the integral as suggested in (Rasmussen, 2000; Neal, 2000).

## 5.2 Heuristic Boundary Elimination

To reduce the inference load on the boundary variables  $b_t$ , we exploit acoustic cues in the feature space to eliminate  $b_t$ 's that are unlikely to be phonetic boundaries. We follow the pre-segmentation method described in Glass (2003) to achieve the goal. For the rest of the boundary variables that are proposed by the heuristic algorithm, we randomly initialize their values and proceed with the sampling process described above.

## 6 Experimental Setup

To the best of our knowledge, there are no standard corpora for evaluating unsupervised methods for acoustic modeling. However, numerous related studies have reported performance on the TIMIT corpus (Dusan and Rabiner, 2006; Estevan et al., 2007; Qiao et al., 2008; Zhang and Glass, 2009; Zhang et al., 2012), which creates a set of strong baselines for us to compare against. Therefore, the TIMIT corpus is chosen as the evaluation set for our model. In this section, we describe the methods used to measure the performance of our model on the following three tasks: sub-word acoustic modeling, segmentation and nonparametric clustering.

**Unsupervised Segmentation** We compare the phonetic boundaries proposed by our model to the manual labels provided in the TIMIT dataset. We follow the suggestion of (Scharenborg et al., 2010) and use a 20-ms tolerance window to compute recall, precision rates and F-score of the segmentation our model proposed for TIMIT's training set. We compare our model against the state-of-the-art unsupervised and semi-supervised segmentation methods that were also evaluated on the TIMIT training set (Dusan and Rabiner, 2006; Qiao et al., 2008).

**Nonparametric Clustering** Our model automatically groups speech segments into different clus-

ters. One question we are interested in answering is whether these learned clusters correlate to English phones. To answer the question, we develop a method to map cluster labels to the phone set in a dataset. We align each cluster label in an utterance to the phone(s) it overlaps with in time by using the boundaries proposed by our model and the manually-labeled ones. When a cluster label overlaps with more than one phone, we align it to the phone with the largest overlap.<sup>4</sup> We compile the alignment results for 3696 training utterances<sup>5</sup> and present a confusion matrix between the learned cluster labels and the 48 phonetic units used in TIMIT (Lee and Hon, 1989).

**Sub-word Acoustic Modeling** Finally, and most importantly, we need to gauge the quality of the learned sub-word acoustic models. In previous work, Varadarajan et al. (2008) and Garcia and Gish (2006) tested their models on a phone recognition task and a term detection task respectively. These two tasks are fair measuring methods, but performance on these tasks depends not only on the learned acoustic models, but also other components such as the label-to-phone transducer in (Varadarajan et al., 2008) and the grapheme model in (Garcia and Gish, 2006). To reduce performance dependencies on components other than the acoustic model, we turn to the task of spoken term detection, which is also the measuring method used in (Jansen and Church, 2011).

We compare our unsupervised acoustic model with three supervised ones: 1) an English triphone model, 2) an English monophone model and 3) a Thai monophone model. The first two were trained on TIMIT, while the Thai monophone model was trained with 32 hour clean read Thai speech from the LOTUS corpus (Kasuriya et al., 2003). All of the three models, as well as ours, used three-state HMMs to model phonetic units. To conduct spoken term detection experiments on the TIMIT dataset, we computed a posteriorigram representation for both training and test feature frames over the

<sup>4</sup>Except when a cluster label is mapped to /vcl/ /b/, /vcl/ /g/ and /vcl/ /d/, where the duration of the release /b/, /g/, /d/ is almost always shorter than the closure /vcl/. In this case, we align the cluster label to both the closure and the release.

<sup>5</sup>The TIMIT training set excluding the sa-type subset.

$\gamma$	$\alpha_b$	$\beta$	$\eta$	$\mu_0$	$\kappa_0$	$\alpha_0$	$\beta_0$
1	0.5	3	3	$\mu^d$	5	3	$3/\lambda^d$

Table 1: The values of the hyperparameters of our model, where  $\mu^d$  and  $\lambda^d$  are the  $d^{\text{th}}$  entry of the mean and the diagonal of the inverse covariance matrix of training data.

HMM states for each of the four models. Ten keywords were randomly selected for the task. For every keyword, spoken examples were extracted from the training set and were searched for in the test set using segmental dynamic time warping (Zhang and Glass, 2009).

In addition to the supervised acoustic models, we also compare our model against the state-of-the-art unsupervised methods for this task (Zhang and Glass, 2009; Zhang et al., 2012). Zhang and Glass (2009) trained a GMM with 50 components to decode posteriorgrams for the feature frames, and Zhang et al. (2012) used a deep Boltzmann machine (DBM) trained with pseudo phone labels generated from an unsupervised GMM to produce a posteriorgram representation. The evaluation metrics they used were: 1) P@N, the average precision of the top N hits, where N is the number of occurrences of each keyword in the test set; 2) EER: the average equal error rate at which the false acceptance rate is equal to the false rejection rate. We also report experimental results using the P@N and EER metrics.

**Hyperparameters and Training Iterations** The values of the hyperparameters of our model are shown in Table 1, where  $\mu^d$  and  $\lambda^d$  are the  $d^{\text{th}}$  entry of the mean and the diagonal of the inverse covariance matrix computed from training data. We pick these values to impose weak priors on our model.<sup>6</sup> We run our sampler for 20,000 iterations, after which the evaluation metrics for our model all converged. In Section 7, we report the performance of our model using the sample from the last iteration.

## 7 Results

Fig. 4 shows a confusion matrix of the 48 phones used in TIMIT and the sub-word units learned from 3696 TIMIT utterances. Each circle represents a mapping pair for a cluster label and an English phone. The confusion matrix demonstrates a strong

<sup>6</sup>In the future, we plan to extend the model and infer the values of these hyperparameters from data directly.

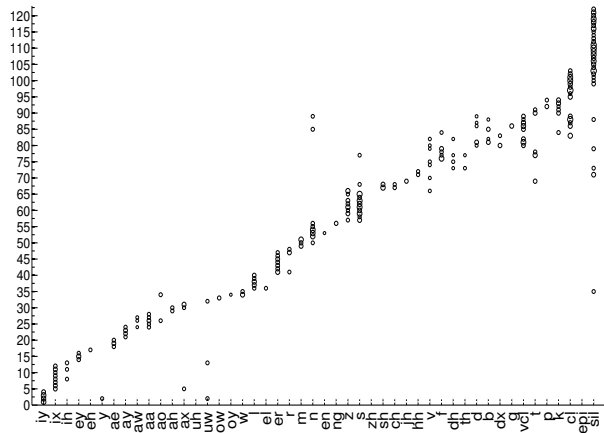


Figure 4: A confusion matrix of the learned cluster labels from the TIMIT training set excluding the sa type utterances and the 48 phones used in TIMIT. Note that for clarity, we show only pairs that occurred more than 200 times in the alignment results. The average co-occurrence frequency of the mapping pairs in this figure is 431.

correlation between the cluster labels and individual English phones. For example, clusters 19, 20 and 21 are mapped exclusively to the vowel /ae/. A more careful examination on the alignment results shows that the three clusters are mapped to the same vowel in a different acoustic context. For example, cluster 19 is mapped to /ae/ followed by stop consonants, while cluster 20 corresponds to /ae/ followed by nasal consonants. This context-dependent relationship is also observed in other English phones and their corresponding sets of clusters. Fig. 4 also shows that a cluster may be mapped to multiple English phones. For instance, clusters 85 and 89 are mapped to more than one phone; nevertheless, a closer look reveals that these clusters are mapped to /n/, /d/ and /b/, which are sounds with a similar place of articulation (i.e. labial and dental). These correlations indicate that our model is able to discover the phonetic composition of a set of speech data without any language-specific knowledge.

The performance of the four acoustic models on the spoken term detection task is presented in Table 2. The English triphone model achieves the best P@N and EER results and performs slightly better than the English monophone model, which indicates a correlation between the quality of an acoustic model and its performance on the spoken term detection task. Although our unsupervised model does not perform as well as the supervised English

unit(%)	P@N	EER
English triphone	75.9	11.7
English monophone	74.0	11.8
Thai monophone	56.6	14.9
Our model	63.0	16.9

Table 2: The performance of our model and three supervised acoustic models on the spoken term detection task.

acoustic models, it generates a comparable EER and a more accurate detection performance for top hits than the Thai monophone model. This indicates that even without supervision, our model captures and learns the acoustic characteristics of a language automatically and is able to produce an acoustic model that outperforms a language-mismatched acoustic model trained with high supervision.

Table 3 shows that our model improves P@N by a large margin and generates only a slightly worse EER than the GMM baseline on the spoken term detection task. At the end of the training process, our model induced 169 HMMs, which were used to compute posteriorgrams. This seems unfair at first glance because Zhang and Glass (2009) only used 50 Gaussians for decoding, and the better result of our model could be a natural outcome of the higher complexity of our model. However, Zhang and Glass (2009) pointed out that using more Gaussian mixtures for their model did not improve their model performance. This indicates that the key reason for the improvement is our joint modeling method instead of simply the higher complexity of our model.

Compared to the DBM baseline, our model produces a higher EER; however, it improves the relative detection precision of top hits by 24.3%. As indicated in (Zhang et al., 2012), the hierarchical structure of DBM allows the model to provide a descent posterior representation of phonetic units. Even though our model only contains simple HMMs and Gaussians, it still achieves a comparable, if not better, performance as the DBM baseline. This demonstrates that even with just a simple model structure, the proposed learning algorithm is able to acquire rich phonetic knowledge from data and generate a fine posterior representation for phonetic units.

Table 4 summarizes the segmentation performance of the baselines, our model and the heuristic

unit(%)	P@N	EER
GMM (Zhang and Glass, 2009)	52.5	16.4
DBM (Zhang et al., 2012)	51.1	14.7
Our model	63.0	16.9

Table 3: The performance of our model and the GMM and DBM baselines on the spoken term detection task.

unit(%)	Recall	Precision	F-score
Dusan (2006)	75.2	66.8	70.8
Qiao et al. (2008)*	77.5	76.3	76.9
Our model	76.2	76.4	76.3
Pre-seg	87.0	50.6	64.0

Table 4: The segmentation performance of the baselines, our model and the heuristic pre-segmentation on TIMIT training set. \*The number of phone boundaries in each utterance was assumed to be known in this model.

pre-segmentation (pre-seg) method. The language-independent pre-seg method is suitable for seeding our model. It eliminates most unlikely boundaries while retaining about 87% true boundaries. Even though this indicates that at best our model only recalls 87% of the true boundaries, the pre-seg reduces the search space significantly. In addition, it also allows the model to capture proper phone durations, which compensates the fact that we do not include any explicit duration modeling mechanisms in our approach. In the best semi-supervised baseline model (Qiao et al., 2008), the number of phone boundaries in an utterance was assumed to be known. Although our model does not incorporate this information, it still achieves a very close F-score. When compared to the baseline in which the number of phone boundaries in each utterance was also unknown (Dusan and Rabiner, 2006), our model outperforms in both recall and precision, improving the relative F-score by 18.8%. The key difference between the two baselines and our method is that our model does not treat segmentation as a stand-alone problem; instead, it jointly learns segmentation, clustering and acoustic units from data. The improvement on the segmentation task shown by our model further supports the strength of the joint learning scheme proposed in this paper.

## 8 Conclusion

We present a Bayesian unsupervised approach to the problem of acoustic modeling. Without any prior

knowledge, this method is able to discover phonetic units that are closely related to English phones, improve upon state-of-the-art unsupervised segmentation method and generate more precise spoken term detection performance on the TIMIT dataset. In the future, we plan to explore phonological context and use more flexible topological structures to model acoustic units within our framework.

## Acknowledgements

The authors would like to thank Hung-an Chang and Ekapol Chuangsuwanich for training the English and Thai acoustic models. Thanks to Matthew Johnson, Ramesh Sridharan, Finale Doshi, S.R.K. Branavan, the MIT Spoken Language Systems group and the anonymous reviewers for helpful comments.

## References

- Chun-An Chan and Lin-Shan Lee. 2011. Unsupervised hidden Markov modeling of spoken queries for spoken term detection without speech recognition. In *Proceedings of INTERSPEECH*, pages 2141 – 2144.
- Steven B. Davis and Paul Mermelstein. 1980. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 28(4):357–366.
- Sorin Dusan and Lawrence Rabiner. 2006. On the relation between maximum spectral transition positions and phone boundaries. In *Proceedings of INTERSPEECH*, pages 1317 – 1320.
- Yago Pereiro Estevan, Vincent Wan, and Odette Scharenborg. 2007. Finding maximum margin segments in speech. In *Proceedings of ICASSP*, pages 937 – 940.
- Emily Fox, Erik B. Sudderth, Michael I. Jordan, and Alan S. Willsky. 2011. A sticky HDP-HMM with application to speaker diarization. *Annals of Applied Statistics*.
- Alvin Garcia and Herbert Gish. 2006. Keyword spotting of arbitrary words using minimal speech resources. In *Proceedings of ICASSP*, pages 949–952.
- John S. Garofolo, Lori F. Lamel, William M. Fisher, Jonathan G. Fiscus, David S. Pallet, Nancy L. Dahlgren, and Victor Zue. 1993. Timit acoustic-phonetic continuous speech corpus.
- Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. 2004. *Bayesian Data Analysis*. Texts in Statistical Science. Chapman & Hall/CRC, second edition.
- James Glass. 2003. A probabilistic framework for segment-based speech recognition. *Computer Speech and Language*, 17:137 – 152.
- Sharon Goldwater. 2009. A Bayesian framework for word segmentation: exploring the effects of context. *Cognition*, 112:21–54.
- Aren Jansen and Kenneth Church. 2011. Towards unsupervised training of speaker independent acoustic models. In *Proceedings of INTERSPEECH*, pages 1693 – 1696.
- Frederick Jelinek. 1976. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64:532 – 556.
- Sawit Kasuriya, Virach Sornlertlamvanich, Patcharika Cotsomrong, Supphanat Kanokphara, and Nattanun Thatphithakkul. 2003. Thai speech corpus for Thai speech recognition. In *Proceedings of Oriental CO-COSDA*, pages 54–61.
- Kai-Fu Lee and Hsiao-Wuen Hon. 1989. Speaker-independent phone recognition using hidden Markov models. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 37:1641 – 1648.
- Chin-Hui Lee, Frank Soong, and Biing-Hwang Juang. 1988. A segment model based approach to speech recognition. In *Proceedings of ICASSP*, pages 501–504.
- Kevin P. Murphy. 2007. Conjugate Bayesian analysis of the Gaussian distribution. Technical report, University of British Columbia.
- Radford M. Neal. 2000. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265.
- Yu Qiao, Naoya Shimomura, and Nobuaki Minematsu. 2008. Unsupervised optimal phoeme segmentation: Objectives, algorithms and comparisons. In *Proceedings of ICASSP*, pages 3989 – 3992.
- Carl Edward Rasmussen. 2000. The infinite Gaussian mixture model. In *Advances in Neural Information Processing Systems*, 12:554–560.
- Odette Scharenborg, Vincent Wan, and Mirjam Ernestus. 2010. Unsupervised speech segmentation: An analysis of the hypothesized phone boundaries. *Journal of the Acoustical Society of America*, 127:1084–1095.
- Balakrishnan Varadarajan, Sanjeev Khudanpur, and Emmanuel Dupoux. 2008. Unsupervised learning of acoustic sub-word units. In *Proceedings of ACL-08: HLT, Short Papers*, pages 165–168.
- Yaodong Zhang and James Glass. 2009. Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams. In *Proceedings of ASRU*, pages 398 – 403.
- Yaodong Zhang, Ruslan Salakhutdinov, Hung-An Chang, and James Glass. 2012. Resource configurable spoken query detection using deep Boltzmann machines. In *Proceedings of ICASSP*, pages 5161–5164.

# Automated Essay Scoring Based on Finite State Transducer: towards ASR Transcription of Oral English Speech

Xingyuan Peng\*, Dengfeng Ke\*, Bo Xu\*

\* Digital Content Technology and Services Research Center

† National Lab of Pattern Recognition

Institute of Automation, Chinese Academy of Sciences

No.95 Zhongguancun East Road, Haidian district, Beijing 100190, China

{xingyuan.peng, dengfeng.ke, xubo}@ia.ac.cn

## Abstract

Conventional Automated Essay Scoring (AES) measures may cause severe problems when directly applied in scoring Automatic Speech Recognition (ASR) transcription as they are error sensitive and unsuitable for the characteristic of ASR transcription. Therefore, we introduce a framework of Finite State Transducer (FST) to avoid the shortcomings. Compared with the Latent Semantic Analysis with Support Vector Regression (LSA-SVR) method (stands for the conventional measures), our FST method shows better performance especially towards the ASR transcription. In addition, we apply the synonyms similarity to expand the FST model. The final scoring performance reaches an acceptable level of 0.80 which is only 0.07 lower than the correlation (0.87) between human raters.

## 1 Introduction

The assessment of learners' language abilities is a significant part in language learning. In conventional assessment, the problem of limited teacher availability has become increasingly serious with the population increase of language learners. Fortunately, with the development of computer techniques and machine learning techniques (natural language processing and automatic speech recognition), Computer-Assisted Language Learning (CALL) systems help people to learn language by themselves.

One form of CALL is evaluating the speech of the learner. Efforts in speech assessment usually fo-

cus on the integrality, fluency, pronunciation, and prosody (Cucchiari et al., 2000; Neumeyer et al., 2000; Maier et al., 2009; Huang et al., 2010) of the speech, which are highly predictable like the exam form of the read-aloud text passage. Another form of CALL is textual assessment. This work is also named AES. Efforts in this area usually focus on the content, arrangement and language usage (Landauer et al., 2003; Ishioka and Kameda, 2004; Kakkonen et al., 2005; Attali and Burstein, 2006; Burstein et al., 2010; Persing et al., 2010; Peng et al., 2010; Attali, 2011; Yannakoudakis et al., 2011) of the text written by the learner under a certain form of examination.

In this paper, our evaluation objects are the oral English picture compositions in English as a Second Language (ESL) examination. This examination requires students to talk about four successive pictures with at least five sentences in one minute, and the beginning sentence is given. This examination form combines both of the two forms described above. Therefore, we need two steps in the scoring task. The first step is Automatic Speech Recognition (ASR), in which we get the speech scoring features as well as the textual transcriptions of the speeches. Then, the second step could grade the text-free transcription in an (conventional) AES system. The present work is mainly about the AES system under the certain situation as the examination grading criterion is more concerned about the integrated content of the speech (the reason will be given in subsection 3.1).

There are many features and techniques which are very powerful in conventional AES systems, but

applying them in this task will cause two different problems as the scoring objects are the ASR output results. The first problem is that the inevitable recognition errors of the ASR will affect the performance of the feature extractions and scoring system. The second problem is caused by the special characteristic of the ASR result. As all these methods are designed under the normal AES situation that they are not suitable for the characteristic.

The impact of the first problem can be reduced by either perfecting the results of the ASR system or building the AES system which is not sensitive to the ASR errors. Improving the performance of the ASR is not what we concern about, so building an error insensitive AES system is what we care about in this paper. This makes many conventional features no longer useful in the AES system, such as spelling errors, punctuation errors and even grammar errors.

The second problem is caused by applying the bag-of-words (BOW) techniques to score the ASR transcription. The BOW are very useful in measuring the content features and are usually robust even if there are some errors in the scoring transcription. However, the robustness would not exist anymore because of the characteristic of the ASR result. It is known that better performance of ASR (reduce the word error rate in ASR) usually requires a strong constrain Language Model (LM). It means that more meaningless parts of the oral speeches would be recognized as the words quite related to the topic content. These words will usually be the key words in the BOW methods, which will lead to a great disturbance for the methods. Therefore, the conventional BOW methods are no longer appropriate because of the characteristic of the ASR result.

To tackle the two problems described above, we apply the FST (Mohri, 2004). As the evaluating objects are from an oral English picture composition examination, it has two important features that make the FST algorithm quite suitable.

- Picture composition examinations require students to speak according to the sequence of the pictures, so there is strong sequentiality in the speech.
- The sentences for describing the same picture are very identical in expression, so there is a hierarchy between the word sequences in the

sentences (the expression) and the sense for the same picture.

FST is designed to describe a structure mapping two different types of information sequences. It is very useful in expressing the sequences and the hierarchy in picture composition. Therefore, we build a FST-based model to extract features related to the transcription assessment in this paper. As the FST-based model is similar to the BOW metrics, it is also an error insensitive model. In this way, the impact of the first problem could be reduced. The FST model is very powerful in delivering the sequence information that a meaningless sequence of words related to the topic content will get low score under the model. Therefore, it works well concerning the second problem. In a word, the FST model can not only be insensitive to the recognition error in the ASR system, but also remedy the weakness of BOW methods in ASR result scoring.

In the remainder of the paper, the related work of conventional AES methods is addressed in section 2. The details of the speech corpus and the examination grading criterion are introduced in section 3. The FST model and its improved method are proposed in section 4. The experiments and the results are presented in section 5. The final section presents the conclusion and future work.

## 2 Related Work

Conventional AES systems usually exploit textual features to assess the quality of writing mainly in three different facets: the content facet, the arrangement facet and the language usage facet. In the content facet, many existing BOW techniques have been applied, such as the content vector analysis (Attali and Burstein, 2006; Attali, 2011) and the LSA to reduce the dimension of content vector (Landauer et al., 2003; Ishioka and Kameda, 2004; Kakkonen et al., 2005; Peng et al., 2010). In arrangement facet, Burstein et al. (2010) modeled the coherence in student essays, while Persing et al. (2010) modeled the organization. In language usage facet, grammar, spelling and punctuation are common features in assessment of the writing competence (Landauer et al., 2003; Attali and Burstein, 2006), and so does the diversity of words and clauses (Lonsdale and Strong-Krause, 2003; Ishioka and Kameda, 2004). Besides

Grading levels		Content Integrity	Acoustic
(18-20)	passed	Describe the information in the four pictures with proper elaboration	Perfect
(15-17)		Describe all the information in all of the four pictures	Good
(12-14)		Describe most of the information in all of the four pictures	Allow errors
(9-11)	failed	Describe most of the information in the pictures, but lose about 1 or 2 pictures	--
(6-8)		Describe some of the information in the pictures, but lose about 2 or 3 pictures	
(3-5)		Describe little information in the four pictures	
(0-2)		Describe some words related to the four pictures	

Table 1: Criterion of Grading

the textual features, many methods are also proposed to evaluate the quality. The cosine similarity is one of the most common used similarity measures (Laudauer et al., 2003; Ishioka and Kameda, 2004; Attali and Burstein, 2006; Attali, 2011). Also, the regression or the classification method is a good choice for scoring (Rudner and Liang, 2002; Peng et al., 2010). The rank preference techniques show excellent performance in grading essays (Yannakoudakis et al., 2011). Chen et al. (2010) proposed an unsupervised approach to AES.

As our work concerns more about the content integrity, we applied the LSA-SVR approach (Peng et al., 2010) as the contrast experiment, which is very effective and robust. In the LSA-SVR method, each essay transcription is represented by a latent semantic space vector, which is regarded as the features in the SVR model. The LSA (Deerwester et al., 1990) considers the relations between the dimensions in conventional vector space model (VSM) (Salton et al., 1975), and it can order the importance of each dimension in the Latent Semantic Space (LSS). Therefore, it is useful in reducing the dimensions of the vector by truncate the high dimensions. The support vector machine can be performed for the function estimation (Smola and Schölkopf, 2004). The LSA-SVR method takes the LSS vector as the feature vector, and applies the SVR for the training data to obtain the SVR model. Each test transcription represented by the LSS vector can be scored by the model.

### 3 Data

As characteristics of the data determine the effectiveness of our methods, the details of it will be introduced first. Our experimental data is acquired in an oral English examination for ESL students. Three

score	> 0	> 12	> 15	> 18
<b>WER(%)</b>	58.86	50.58	45.56	36.36
<b>MR(%)</b>	72.88	74.03	75.70	78.45

Table 2: WER and MR of ASR result

classes of students participated in the exam and 417 valid speeches are obtained in the examination. As the paper mainly focuses on scoring the text transcriptions, we have two ways to obtain them. One is manually typing the text transcriptions which we regarded as the Correct Recognition Result (CRR) transcription, and another is the ASR result which we named ASR transcription. We use the HTK (Young et al., 2006), which stands for the state of art in speech recognition, to build the ASR system.

To better reveal the differences of the methods' performance, all the experiments will be done in both transcriptions. A better understanding of the difference in the CRR transcription and the ASR transcription from the low score to the high score is shown in Table 2, where WER is the word error rate and MR is the match rate which is the words' correct rate.

#### 3.1 Criterion of Grading

According to the Grading Criterion of the examination, the score of the examination ranges from 0 to 20, and the grading score is divided into 7 levels with 3 points' interval for each level. The criterion mainly concerns about two facets of the speech: the acoustic level and the content integrity. The details of the criterion are shown in Table 1. The criterion indicates that the integrity is the most important part in rating the speech. The acoustic level only works well in excellent speeches (Huang et al., 2010). Therefore, this paper mainly focuses on the integrity



Correlation	R1	R2	R3	ES	OC
<b>R1</b>	-	0.8966	0.8557	0.9620	0.9116
<b>R2</b>	-	-	0.8461	0.9569	0.9048
<b>R3</b>	-	-	-	0.9441	0.8739
<b>Average</b>		0.8661		0.9543	0.8968

Table 3: Correlations of Human Scores

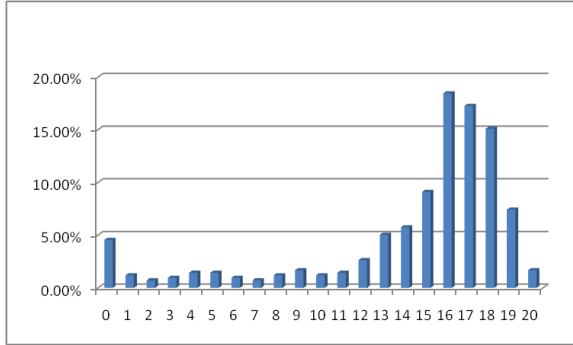


Figure 1: Distribution of Final Expert Scores

of content. The acoustic level as well as other levels such as grammar errors is ignored. Because the criterion is almost based on the content, our methods obtain good performance although we ignore some features.

### 3.2 Human Score Correlation and Distribution

Each speech in our experiments was scored by three raters. Therefore, we have three scores for each speech. The final expert score is the average of these three scores. The correlations between human scores are shown in Table 3.

R1, R2, and R3 stand for the three raters, and ES is the final expert score. The Open Correlation (OC) is the correlation between human rater scores and the final scores, which are not related to the human scores themselves (average of the other two scores).

As most students are supposed to pass the examination, the expert scores are mostly distributed above 12 points, as shown in Figure 1. In the range of the pass score, the distribution is close to normal distribution, while in the range of failed score except 0, the distribution is close to uniform distribution.

## 4 Approach

The approach used in this paper is to build a standard FST for the current examination topic. However, the annotation of the corpus is necessary before the

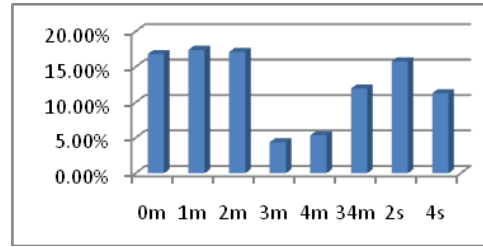


Figure 2: Distribution of Sentence Labels

building. After the annotation and the building, the features are extracted based on the FST. The automated machine score is computed from the features at last. Therefore, subsection 4.1 will show the corpus annotation, subsection 4.2 will introduce how to build the standard FST of the current topic, and subsections 4.3 and 4.4 will discuss how to extract the features, at last, an improved method is proposed in subsection 4.5.

### 4.1 Corpus Annotation

The definitions of the sequences and hierarchy in the corpus will be given before we apply the FST algorithm. According to the characteristics of the picture composition examination, each composition can be held as an orderly combination of the senses of pictures. The senses of pictures are called sense-groups here. We define a sense-group as one sentence either describing the same one or two pictures or elaborating on the same pictures. The description sentence is labeled with a tag 'm'(main sense of the picture) and the elaboration one is labeled with 's'(subordinate sense of the picture). The first given sentence in the examination is labeled with 0m and the other describing sentences for the 1 to 4 pictures are labeled with 1m to 4m, while the elaboration ones for the 4 pictures are labeled with 1s to 4s. Therefore, each sentence in the composition is labeled as a sense-group. For the entire 417 CRR transcriptions, we manually labeled 274 transcriptions whose scores are higher than 15 points. We gained 8 types of labels from the manually labeled results. They are 0m, 1m, 2m, 3m, 34m (one sentence describes both of the third and the fourth pictures), 4m, 2s and 4s. Other labels were discarded for the number of their appearance is very low. The distribution of sentences with each label is shown in Figure 2. There are 1679 sentences in the 274 CRR

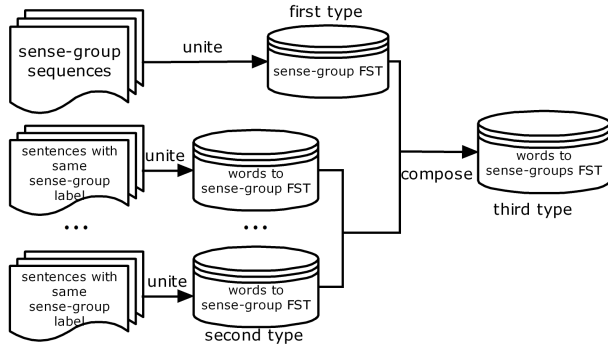


Figure 3: FST Building

transcriptions and 1667 are labeled in the eight symbols.

## 4.2 FST Building

In this paper, we build three types of FST to extract scoring features with the help of openFST tool (Al-lauzen et al., 2007). The first is the sense-group FST, the second is the words to each sense-group FST and the last is the words to all the sense-groups FST. They are shown in Figure 3.

The definition of the sense-group has been given in subsection 4.1. The sense-group FST can describe all the possible proper sense-group sequences of the current picture composition topic. It is also an acceptor trained from the labeled corpus. We use manually labeled corpus, which are the sequences of sense-groups of the CRR transcriptions with expert scores higher than 15 points, to build the sense-group FST. In the process, each CRR transcription sense-group sequence is a simple sense-group FST. Later, we unite these sense-group FSTs to get the final FST which considers every situation of sense-group sequences in the train corpus. Also, we use the operation of "determinize" and "minimize" in openFST to optimize the final sense-group FST that its states have no same input label and is a smallest FST.

The second type is the words to sense-group FST. It determines what word sequence input will result in what sense-group output. With the help of these FSTs, we can find out how students use language to describe a certain sense-group, or in other words, a certain sense-group is usually constructed with what kind of word sequence. All the different sentences with their sense-group labels are tak-

en from the train corpus. We regard each sentence as a simple words to sense-group FST, and then unite these FSTs which have the same sense-group label. The final union FSTs can transform proper word sequence into the right sense-group. Like building the sense-group FST, the optimization operations of "determinize" and "minimize" are also done for the FSTs.

The last type of FST is a words to sense-groups FST. We can also treat it as a words FSA, because any word sequence accepted by the words to sense-groups FST is considered to be an integrated composition. Meanwhile, it can transform the word sequence into the sense-group label sequence which is very useful in extracting the scoring features (details will be presented in subsection 4.4). The FST is built from the other two types of FST that we made before. We compute the composition of all the words to each sense-group FSTs (the second type) and the sense-group FST (the first type) with the operations of "compose" in openFST. Then, the composition result is the words to sense-groups FST, the third type of FST in this paper.

## 4.3 Search for the Best Path in FST

Now we have successfully built the words to sense-groups FST, the third type described above. Just like the similarity methods mentioned in section 2 can score essays from a have-been-scored similar essay, we need to find the best path, which is closest to the to-be-scored transcription, in the FST. Here, we apply the edit distance to measure how best the path is. This means the best path is the word sequence path in the FST which has the smallest edit distance compared with the to-be-scored transcription's word sequences.

Here, we modify the Wagner-Fischer algorithm (Wagner and Fischer, 1974), which is a Dynamic Programming (DP) algorithm, to quest the best path in the FST. A simple example is illustrated in Figure 4. The best path can be described as

$$path = \arg \min_{\substack{path \in \\ allpath}} EDcost(path, transcription) \quad (1)$$

$$EDcost = ins + del + sub \quad (2)$$

EDcost is the edit distance from the transcription to the paths which start at state 0 and end at the end

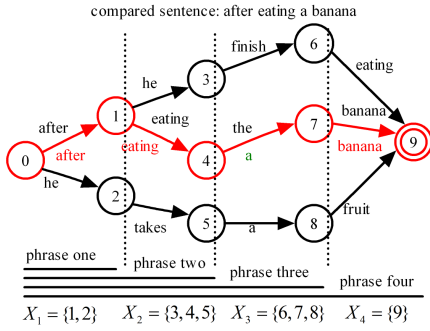


Figure 4: Search the Best Path in the FST by DP

state. The DP process can be described by equation (3):

$$\min EDcost(i) = \arg \min_{j \in X_1, \dots, X_{p-1}} (\min EDcost(j) + cost(j, i)) \quad (3)$$

The  $\min EDcost(j)$  is the accumulated minimum edit distance from state 0 to state  $j$ , and the  $cost(i, j)$  is the cost of insertion, deletion or substitution from state  $j$  to state  $i$ . The equation means the  $\min ED$  of state  $i$  can be computed by the accumulated  $\min ED$ -cost of state  $j$  in the phase  $p$ . The state  $j$  belongs to the have-been-calculated state set  $\{X_0, \dots, X_{p-1}\}$  in phase  $p$ . In phrase  $p$ , we compute the best path and its edit distance from the transcription for all the to-be-calculated states which is the  $X_p$  shown in Figure 4. After computing all the phrases, the best path and its edit distances of the end states are obtained. Then the final best path is the one with the smallest edit distance.

#### 4.4 Feature Extraction

After building the FST and finding the best path for the to-be-scored transcription, we can extract some effective features from the path information and the transcription. Inspired by the similarity scoring measures, our proposed features represent the similarity between the best path's word sequence and the to-be-scored transcription.

The features used for the scoring model are as follows:

- **The Edit Distance (ED):**

The edit distance is the linear combination of the weights of insertion, deletion and substitution. The relation is shown in equation (2), where  $ins$ ,  $del$  and  $sub$  are the appearance times

of insertions, deletions and substitutions, respectively. Normally, we set the cost of each to be 1.

- **The Normalized Edit Distance(NED):**

The NED is the ED normalized with the transcription's length.

$$NEDcost = EDcost/length \quad (4)$$

- **The Match Number(MN):**

The match number is the number of words matched between the best path and the transcription.

- **The Match Rate(MR):**

The match rate is the match number normalized with the transcription's length.

$$MR = MN/length \quad (5)$$

- **The Continuous Match Value(CMV):**

Continuous match should be better than the fragmentary match, so a higher value is given for the continuous situation.

$$CMV = \sum OM + 2\sum SM + 3\sum LM \quad (6)$$

where OM (One Match) is the fragmentary match number, SM (Short Match) is the continuous match number which is no more than 4, and LM (Long Match) is the continuous match number which is more than 4.

- **The Length(L):**

The length of transcription. Length is always a very effective feature in essay scoring (Attali and Burstein, 2006).

- **The Sense-group Scoring Feature(SSF):**

For each best path, we can transform the transcription's word sequence into the sense-group label sequence with the FST. Then, the words match rate of each sense-group can be computed. The match rate of each sense-group can be regarded as one feature so that all the sense-group match rate in the transcription will be combined to a feature vector (called the Sense-group Match Rate vector (SMRv)), which is an 8-dimensional vector in the present experiments. After that, we applied the SVR algorithm to train a sense-group scoring model with the vectors and scores, and the transcription gets its SSF from the model.

#### 4.5 Extend the FST model with the similarity of synonym

Because the FST is trained from the limited corpus, it does not contain all the possible situations proper for the current composition topic. To complete the current FST model, we add the similarity of synonym to extend the FST model so that it can handle more situations.

The extension of the FST model is mainly reflected in calculation of the edit distance of the best path. The previous edit distance, in equation (2), refers to the Levenshtein distance in which the insertions, deletions and substitutions have equal cost, but in the edit distance in this section, the cost of substitutions is less than that of insertions and deletions. Here, we assume that the cost of substitutions is based on the similarity of the two words. Then with the help of different cost of substitutions, each word edge is extended to some of its synonym word edges under the cost of similarity. The new edit distance is calculated by equation (7) as follows:

$$ED_{cost} = ins + del + sub \times (1 - sim) \quad (7)$$

where,  $sim$  is the similarity of two words.

We used the Wordnet::Similarity software package (Pedersen et al., 2004) to calculate the similarity between every two words at first. However, the performance's reduction of the AES system indicates that the similarity is not good enough to extend the FST model. Therefore, we seek for human help to accurate the similarity calculation. We manually checked the similarity, and deleted some improper similarity. Thus the final similarity applied in our experiment is the Wordnet::Similarity software computing result after the manual check.

## 5 Experiments

In this section, the proposed features and our FST methods will be evaluated on the corpus we mentioned above. The contrasting approach, the LSA-SVR approach, will also be presented.

### 5.1 Data Setup

The experiment corpus consists of 417 speeches. With the help of manual typing and the ASR system, 417 CRR transcriptions and 417 ASR transcriptions are obtained from the speeches after preprocessing

FST build	SVR train	SVR test	CRR transcription	ASR transcription
Set2	Set3	Set1	0.7999	0.7505
Set3	Set2		0.8185	0.7401
Set1	Set3	Set2	0.8557	0.7372
Set3	Set1		0.8111	0.7257
Set1	Set2	Set3	0.9085	0.8086
Set2	Set1		0.8860	0.8086

Table 4: Correlation Between the SSF and the Expert Scores

which includes the capitalization processing and the stemming processing. We divide them into 3 sets by the same distribution of their scores. Therefore, there are totally 6 sets, and each of them has 139 of the transcriptions. The FST building only uses the CRR transcriptions whose expert scores are higher than 15 points. While treating one set (one CRR set) as the FST building train set, we get the ED, NED, MN, MR, CMV features and the SMR vectors for the other two sets (could be either CRR sets or ASR sets). Then, the SSF is obtained by another set as the SVR train set and the last set as the test set. The parameters of the SVR are trained through the grid search from the whole data sets (ASR or CRR sets) by cross-validation. Therefore, except the length feature, the other six features of each set can be extracted from the FST model.

Also, we presented the result of using LSA-SVR approach as a contrast experiment to show the improvement of our FST model in scoring oral English picture composition.

To quantitatively assess the effectiveness of the methods, the Pearson correlation between the expert scores and the automated results is adopted as the performance measure.

### 5.2 Correlation of Features

The correlations between the seven features and the final expert scores are shown in Tables 4 and 5 on the three sets.

The MN and CMV are very good features, while the NED is not. This is mainly due to the nature of the examination. When scoring the speech, human raters concern more about how much valid information it contains and irrelevant contents are not taken for penalty. Therefore, the match features are more reasonable than the edit distance features. This im-

Script	Train	Test	L	ED	NED	MN	MR	CMV
CRR	Set2	Set1	0.7404	0.2410	-0.6690	0.8136	0.1544	0.7417
	Set3			0.3900	-0.4379	0.8316	0.1386	0.7792
	Set1	Set2	0.7819	0.4029	-0.7667	0.8205	0.4904	0.7333
	Set3			0.4299	-0.5672	0.8370	0.5090	0.7872
	Set1	Set3	0.8645	0.4983	-0.7634	0.8867	0.2718	0.8162
	Set2			0.3639	-0.6616	0.8857	0.3305	0.8035
Average			0.7956	0.3877	-0.6443	0.8459	0.3158	0.7769
ASR	Set2	Set1	0.1341	-0.2281	-0.6375	0.7306	0.6497	0.7012
	Set3			-0.1633	-0.5110	0.7240	0.6071	0.6856
	Set1	Set2	0.2624	-0.0075	-0.4640	0.6717	0.5929	0.6255
	Set3			0.0294	-0.4389	0.6860	0.6259	0.6255
	Set1	Set3	0.1643	-0.1871	-0.5391	0.7419	0.6213	0.7001
	Set2			-0.1742	-0.4721	0.7714	0.6199	0.7329
Average			0.1869	-0.1218	-0.5104	0.7209	0.6195	0.6785

Table 5: Correlations Between the Six Features and the Expert Scores

Script Method	Set1	Set2	Set3	Average
Length	0.7404	0.7819	0.8645	0.7956
CRR LSA-SVR	0.7476	0.8024	0.8663	0.8054
FST	0.8702	0.8852	0.9386	<b>0.8980</b>
Length	0.1341	0.2624	0.1643	0.1869
ASR LSA-SVR	0.5975	0.5643	0.5907	0.5842
FST	0.7992	0.7678	0.8452	<b>0.8041</b>

Table 6: Performance of the FST Method, the LSA-SVR Approach and the Length Feature

fact is similar to the result displayed by the ASR output performance in Table 2 in section 3, where the WER has significant difference from the low score speeches to the high score ones while the MR does not, and the MR is much better than the WER.

As the length feature is a strong correlation feature in CRR transcription, the MR feature, which is normalized by the length, is strongly affected. However, with the impact declining in the ASR transcription, the MR feature performs very well. This also explains the reason of different correlations of ED and NED in CRR transcription.

The SSF is entirely based on the FST model, so the impact of the length feature is very low. The decline of it in different transcriptions is mainly because of the ASR error.

### 5.3 Performance of the FST Model

For each test transcription, it has 12 dimensions of FST features. The ED, NED, MN, MR and CMV features have two dimensions of each as trained from

two different FST building sets. The SSF needs two train sets as there are two train models: one is for the FST building model and another is for the SVR model. As different sets for different models, it also has two dimension features. We use the linear regression to combine these 12 features to the final automated score. The linear regression parameters were trained from all the data by cross-validation. After the weight of each feature and the linear bias are gained, we calculate the automated score of each transcription by the FST features. The performance of our FST model is shown in Table 6. Compared with it, the performance of the LSA-SVR algorithm, the baseline in our paper, is also shown. As a usual best feature for AES, the length shows its outstanding performance in CRR transcription. However, it fails in the ASR transcription.

As we have predicted above, the BOW algorithm (the LSA-SVR) performance declines drastically in the ASR transcription, which also happens to the length feature. By contrast, the decline of the performance of our FST method is acceptable considering the impact of recognition errors in the ASR system. This means the FST model is an error insensitive model that is very appropriate for the task.

### 5.4 Improvement of FST by Adding the Similarity

The improved FST extends the original FST model by considering the word similarity in substitutions. In the extension, the similarities of the synonyms

Script	Method	Set1	Set2	Set3	Average
CRR	FST	0.8702	0.8852	0.9386	0.8980
	IFST	0.8788	0.8983	0.9418	0.9063
ASR	FST	0.7992	0.7678	0.8452	0.8041
	IFST	0.8351	0.7617	0.8168	0.8045

Table 7: Performance of the FST Method and the Improved FST Method

describe the invisible (extended) part of the FST, so it should be very accurate for the substitutions cost. Therefore, we added manual intervention to the similarity result calculated by the wordnet::similarity software packet.

After we added the similarity of synonym to extend the FST model, the performance of the new model increased stably in the CRR transcription. However, the increase is not significant in the ASR transcription (shown in Table 7). We believe it is because the superiority of the improved model is disguised by the ASR error. In other words, the impact of ASR error under the FST model is more significant than the improvement of the FST model. The performance correlation of our FST model in the CRR transcription is about 0.9 which is very close to the human raters' (shown in Table 3). Even though the performance correlation in the ASR transcription declines compared with that in the CRR transcription, the FST methods still perform very well under the current recognition errors of the ARS system.

## 6 Conclusion and Future work

The aforementioned experiments indicate three points. First, the BOW algorithm has its own weakness. In regular text essay scoring, the BOW algorithm can have excellent performance. However, in certain situations, such as towards ASR transcription of oral English speech, its weakness of sequence neglect will be magnified, leading to drastic decline of performance. Second, the introduced FST model is suitable in our task. It is an error insensitive model under the task of automated oral English picture composition scoring. Also, it considers the sequence and the hierarchy information. As we expected, the performance of the FST model is more outstanding than that of the BOW metrics in CRR transcription, and the decline of performance is acceptable in ASR transcription scoring. Third, adding the similarity

of synonyms to extend the FST model improves the system performance. The extension can complete the FST model, and achieve better performance in the CRR transcription.

The future work may focus on three facets. First, as the extension of the FST model is a preliminary study, there is much work that can be done, such as calculating the similarity more accurately without manual intervention, or finding a balance between the original FST model and the extended one to improve the performance in ASR transcription. Second, as the task is speech evaluation, considering the acoustic features may give more information to the automated scoring system. Therefore, the features at the acoustic level could be introduced to complete the scoring model. Third, the decline of the performance in ASR transcription is derived from the recognition error of ASR system. Therefore, improving the performance of the ASR system or making full use of the N-best lists may give more accurate transcription for the AES system.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 90820303 and No. 61103152). We thank the anonymous reviewers for their insightful comments.

## References

- Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut and Mehryar Mohri. 2007. OpenFst: a general and efficient weighted finite-state transducer library. In *Proceedings of International Conference on Implementation and Application of Automata*, 4783: 11-23.
- Yigal Attali. 2011. A differential word use measure for content analysis in automated essay scoring. ETS research report, ETS RR-11-36.
- Yigal Attali and Jill Burstein. 2006. Automated essay scoring with e-rater®V.2. *The Journal of Technology, Learning, and Assessment*, 4(3), 1-34.
- Jill Burstein, Joel Tetreault, and Slava Andreyev. 2010. Using entity-based features to model coherence in student essays. In *Human Language Technologies: The Annual Conference of the North American Chapter of the ACL*, 681-684.
- Chih-Chung Chang, Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, Vol. 2.

- Yen-Yu Chen, Chien-Liang Liu, Chia-Hoang Lee, and Tao-Hsing Chang. 2010. An unsupervised automated essay scoring system. *IEEE Intelligent Systems*, 61-67.
- Catia Cucchiarini, Helmer Strik, and Lou Boves. 2000. Quantitative assessment of second language learners' fluency by means of automatic speech recognition technology. *Acoustical Society of America*, 107(2): 989-999.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*. 41(6): 391-407.
- Shen Huang, Hongyan Li, Shijin Wang, Jiaen Liang and Bo Xu. 2010. Automatic reference independent evaluation of prosody quality using multiple knowledge fusions. In *INTERSPEECH*, 610-613.
- Tsunenori Ishioka and Masayuki Kameda. 2004. Automated Japanese essay scoring system: jess. In Proceedings of the International Workshop on database and Expert Systems applications.
- Tuomo Kakkonen, Niko Myller, Jari Timonen, and Erkki Sutinen. 2005. Automatic essay grading with probabilistic latent semantic analysis. In Proceedings of *Workshop on Building Educational Applications Using NLP*, 29-36.
- Thomas K. Landauer, Darrell Laham and Peter Foltz. 2003. Automatic essay assessment. *Assessment in Education: Principles, Policy and Practice* (10:3), 295-309.
- Deryle Lonsdale and Diane Strong-Krause. 2003. Automated rating of ESL essays. In Proceedings of the *HLT-NAACL Workshop: Building Educational Applications Using NLP*.
- Andreas Maier, F. Höning, V. Zeissler, Anton Batliner, E. Körner, N. Yamanaka, P. Ackermann, Elmar Nöth. 2009. A language-independent feature set for the automatic evaluation of prosody. In *INTERSPEECH*, 600-603.
- Mehryar Mohri. 2004. Weighted finite-state transducer algorithms: an overview. *Formal Languages and Applications*, 148 (620): 551-564.
- Leonardo Neumeyer, Horacio Franco, Vassilios Digalakis, Mitchel Weintraub. 2000. Automatic scoring of pronunciation quality. *Speech Communication*, 30(2-3): 83-94.
- Ted Pedersen, Siddharth Patwardhan and Jason Mitchell. 2004. WordNet::Similarity - measuring the relatedness of concepts. In Proceedings of the *National Conference on Artificial Intelligence*, 144-152.
- Xingyuan Peng, Dengfeng Ke, Zhenbiao Chen and Bo Xu. 2010. Automated Chinese essay scoring using vector space models. In Proceedings of *IUCS*, 149-153.
- Isaac Persing, Alan Davis and Vincent Ng. 2010. Modeling organization in student essays. In Proceedings of *EMNLP*, 229-239.
- Lawrence M. Rudner and Tahung Liang. 2002. Automated essay scoring using Bayes' theorem. *The Journal of Technology, Learning and Assessment*, 1(2):3-21.
- G. Salton, C. Yang, A. Wong. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11): 613-620.
- Alex J. Smola and Bernhard Schölkopf. 2004. A tutorial on support vector regression. *Statistics and Computing* 14(3): 199-222.
- Robert A. Wagner and Michael J. Fischer. 1974. The string-to-string correction problem. *Journal of the ACM*, 21(1):168-173.
- Helen Yannakoudakis, Ted Briscoe and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In Proceedings of *ACL*, 180-189.
- Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Valtcho Valtchev, Phil Woodland. 2006. The HTK book (for HTK version 3.4). Cambridge University Engineering Department.

# Text-level Discourse Parsing with Rich Linguistic Features

**Vanessa Wei Feng**

Department of Computer Science  
University of Toronto  
Toronto, ON, M5S 3G4, Canada  
weifeng@cs.toronto.edu

**Graeme Hirst**

Department of Computer Science  
University of Toronto  
Toronto, ON, M5S 3G4, Canada  
gh@cs.toronto.edu

## Abstract

In this paper, we develop an RST-style text-level discourse parser, based on the HILDA discourse parser (Hernault et al., 2010b). We significantly improve its tree-building step by incorporating our own rich linguistic features. We also analyze the difficulty of extending traditional sentence-level discourse parsing to text-level parsing by comparing discourse-parsing performance under different discourse conditions.

## 1 Introduction

In a well-written text, no unit of the text is completely isolated; interpretation requires understanding the unit’s relation with the context. Research in discourse parsing aims to unmask such relations in text, which is helpful for many downstream applications such as summarization, information retrieval, and question answering.

However, most existing discourse parsers operate on individual sentences alone, whereas discourse parsing is more powerful for text-level analysis. Therefore, in this work, we aim to develop a text-level discourse parser. We follow the framework of Rhetorical Structure Theory (Mann and Thompson, 1988) and we take the HILDA discourse parser (Hernault et al., 2010b) as the basis of our work, because it is the first fully implemented text-level discourse parser with state-of-the-art performance. We significantly improve the performance of HILDA’s tree-building step (introduced in Section 5.1 below) by incorporating rich linguistic features (Section 5.3). In our experiments (Section 6), we also analyze the

difficulty with extending traditional sentence-level discourse parsing to text-level parsing, by comparing discourse parsing performance under different discourse conditions.

## 2 Discourse-annotated corpora

### 2.1 The RST Discourse Treebank

Rhetorical Structure Theory (Mann and Thompson, 1988) is one of the most widely accepted frameworks for discourse analysis. In the framework of RST, a coherent text can be represented as a discourse tree whose leaves are non-overlapping text spans called *elementary discourse units* (EDUs); these are the minimal text units of discourse trees. Adjacent nodes can be related through particular discourse relations to form a discourse subtree, which can then be related to other adjacent nodes in the tree structure. According to RST, there are two types of discourse relations, hypotactic (“mononuclear”) and paratactic (“multi-nuclear”). In mononuclear relations, one of the text spans, the *nucleus*, is more salient than the other, the *satellite*, while in multi-nuclear relations, all text spans are equally important for interpretation.

The example text fragment shown in Figure 1 consists of four EDUs ( $e_1$ - $e_4$ ), segmented by square brackets. Its discourse tree representation is shown below in the figure, following the notational convention of RST. The two EDUs  $e_1$  and  $e_2$  are related by a mononuclear relation *ATTRIBUTION*, where  $e_1$  is the more salient span; the span ( $e_1$ - $e_2$ ) and the EDU  $e_3$  are related by a multi-nuclear relation *SAME-UNIT*, where they are equally salient.



[Catching up with commercial competitors in retail banking and financial services,] $e_1$  [they argue,] $e_2$  [will be difficult,] $e_3$  [particularly if market conditions turn sour.] $e_4$

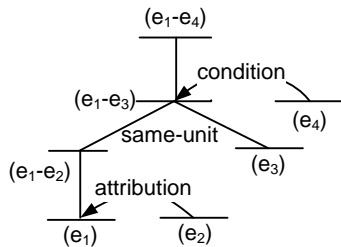


Figure 1: An example text fragment (wsj\_0616) composed of four EDUs, and its RST discourse tree representation.

The RST Discourse Treebank (RST-DT) (Carlson et al., 2001), is a corpus annotated in the framework of RST. It consists of 385 documents (347 for training and 38 for testing) from the *Wall Street Journal*. In RST-DT, the original 24 discourse relations defined by Mann and Thompson (1988) are further divided into a set of 18 relation classes with 78 finer-grained rhetorical relations in total, which provides a high level of expressivity.

## 2.2 The Penn Discourse Treebank

The Penn Discourse Treebank (PDTB) (Prasad et al., 2008) is another annotated discourse corpus. Its text is a superset of that of RST-DT (2159 *Wall Street Journal* articles). Unlike RST-DT, PDTB does not follow the framework of RST; rather, it follows a lexically grounded, predicate-argument approach with a different set of predefined discourse relations, as proposed by Webber (2004). In this framework, a discourse connective (e.g., *because*) is considered to be a predicate that takes two text spans as its arguments. The argument that the discourse connective structurally attaches to is called **Arg2**, and the other argument is called **Arg1** — unlike in RST, the two arguments are not distinguished by their saliency for interpretation. Another important difference between PDTB and RST-DT is that in PDTB, there does not necessarily exist a tree structure covering the full text, i.e., PDTB-styled discourse relations exist only in a very local contextual window. In PDTB, relation types are organized hierarchically: there are 4 *classes*, which can be further divided into 16 *types* and 23 *subtypes*.

## 3 Related work

Discourse parsing was first brought to prominence by Marcu (1997). Since then, many different algorithms and systems (Soricut and Marcu, 2003; Reitter, 2003; LeThanh et al., 2004; Baldrige and Lascarides, 2005; Subba and Di Eugenio, 2009; Sagae, 2009; Hernault et al., 2010b) have been proposed, which extracted different textual information and adopted various approaches for discourse tree building. Here we briefly review two fully implemented text-level discourse parsers with the state-of-the-art performance.

The HILDA discourse parser of Hernault and his colleagues (duVerle and Prendinger, 2009; Hernault et al., 2010b) is the first fully-implemented feature-based discourse parser that works at the full text level. Hernault et al. extracted a variety of lexical and syntactic features from the input text, and trained their system on RST-DT. While some of their features were inspired by the previous work of others, e.g., lexico-syntactic features borrowed from Soricut and Marcu (2003), Hernault et al. also proposed the novel idea of discourse tree building by using two classifiers in cascade — a binary structure classifier to determine whether two adjacent text units should be merged to form a new subtree, and a multi-class classifier to determine which discourse relation label should be assigned to the new subtree — instead of the more-usual single multi-class classifier with the additional label NO-REL. Hernault et al. obtained 93.8% *F*-score for EDU segmentation, 85.0% accuracy for structure classification, and 66.8% accuracy for 18-class relation classification.

Lin et al. (2009) attempted to recognize implicit discourse relations (discourse relations which are not signaled by explicit connectives) in PDTB by using four classes of features — contextual features, constituent parse features, dependency parse features, and lexical features — and explored their individual influence on performance. They showed that the production rules extracted from constituent parse trees are the most effective features, while contextual features are the weakest. Subsequently, they fully implemented an end-to-end PDTB-style discourse parser (Lin et al., 2010).

Recently, Hernault et al. (2010a) argued that more effort should be focused on improving performance

on certain infrequent relations presented in the discourse corpora, since due to the imbalanced distribution of different discourse relations in both RST-DT and PDTB, the overall accuracy score can be overwhelmed by good performance on the small subset of frequent relations, even though the algorithms perform poorly on all other relations. However, because of infrequent relations for which we do not have sufficient instances for training, many unseen features occur in the test data, resulting in poor test performance. Therefore, Hernault et al. proposed a semi-supervised method that exploits abundant, freely-available unlabeled data as a basis for feature vector extension to alleviate such issues.

#### 4 Text-level discourse parsing

Not until recently has discourse parsing for full texts been a research focus — previously, discourse parsing was only performed on the sentence level<sup>1</sup>. In this section, we explain why we believe text-level discourse parsing is crucial.

Unlike syntactic parsing, where we are almost never interested in parsing above sentence level, sentence-level parsing is not sufficient for discourse parsing. While a sequence of local (sentence-level) grammaticality can be considered to be global grammaticality, a sequence of local discourse coherence does not necessarily form a globally coherent text. For example, the text shown in Figure 2 contains two sentences, each of which is coherent and sensible itself. However, there is no reasonable content transition between these two sentences, so the combination of the two sentences does not make much sense. If we attempt to represent the text as an RST discourse tree like the one shown in Figure 1, we find that no discourse relation can be assigned to relate the spans  $(e_1-e_2)$  and  $(e_3-e_4)$  and the text cannot be represented by a valid discourse tree structure.

In order to rule out such unreasonable transitions between sentences, we have to expand the text units upon which discourse parsing is performed: from sentences to paragraphs, and finally paragraphs to

<sup>1</sup>Strictly speaking, for PDTB-style discourse parsing (e.g., Lin et al. (2009; 2010)), there is no absolute distinction between sentence-level and text-level parsing, since in PDTB, discourse relations are annotated at a level no higher than that of adjacent sentences. Here we are concerned with RST-style discourse parsing.

[No wonder he got an A for his English class,] $e_1$  [he was studying so hard.] $e_2$  [He avoids eating chocolates,] $e_3$  [since he is really worried about gaining weight.] $e_4$

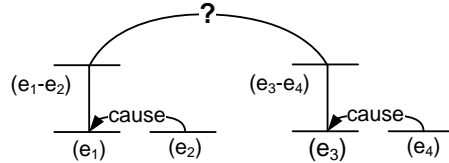


Figure 2: An example of incoherent text fragment composed of two sentences. The two EDUs associated with each sentence are coherent themselves, whereas the combination of the two sentences is not coherent at the sentence boundary. No discourse relation can be associated with the spans  $(e_1-e_2)$  and  $(e_3-e_4)$ .

the full text.

Text-level discourse parsing imposes more constraints on the global coherence than sentence-level discourse parsing. However, if, technically speaking, text-level discourse parsing were no more difficult than sentence-level parsing, any sentence-level discourse parser could be easily upgraded to a text-level discourse parser just by applying it to full texts. In our experiments (Section 6), we show that when applied above the sentence level, the performance of discourse parsing is consistently inferior to that within individual sentences, and we will briefly discuss what the key difficulties with extending sentence-level to text-level discourse parsing are.

#### 5 Method

We use the HILDA discourse parser of Hernault et al. (2010b) as the basis of our work. We refine Hernault et al.’s original feature set by incorporating our own features as well as some adapted from Lin et al. (2009). We choose HILDA because it is a fully implemented text-level discourse parser with the best reported performance up to now. On the other hand, we also follow the work of Lin et al. (2009), because their features can be good supplements to those used by HILDA, even though Lin et al.’s work was based on PDTB. More importantly, Lin et al.’s strategy of performing feature selection prior to classification proves to be effective in reducing the total feature dimensions, which is favorable since we wish to incorporate rich linguistic features into our discourse parser.

### 5.1 Bottom-up approach and two-stage labeling step

Following the methodology of HILDA, an input text is first segmented into EDUs. Then, from the EDUs, a bottom-up approach is applied to build a discourse tree for the full text. Initially, a binary *Structure* classifier evaluates whether a discourse relation is likely to hold between consecutive EDUs. The two EDUs which are most probably connected by a discourse relation are merged into a discourse subtree of two EDUs. A multi-class *Relation* classifier evaluates which discourse relation label should be assigned to this new subtree. Next, the *Structure* classifier and the *Relation* classifier are employed in cascade to re-evaluate which relations are the most likely to hold between adjacent spans (discourse subtrees of any size, including atomic EDUs). This procedure is repeated until all spans are merged, and a discourse tree covering the full text is therefore produced.

Since EDU boundaries are highly correlated with the syntactic structures embedded in the sentences, EDU segmentation is a relatively trivial step — using machine-generated syntactic parse trees, HILDA achieves an *F*-score of 93.8% for EDU segmentation. Therefore, our work is focused on the tree-building step, i.e., the *Structure* and the *Relation* classifiers. In our experiments, we improve the overall performance of these two classifiers by incorporating rich linguistic features, together with appropriate feature selection. We also explore how these two classifiers perform differently under different discourse conditions.

### 5.2 Instance extraction

Because HILDA adopts a bottom-up approach for discourse tree building, errors produced on lower levels will certainly propagate to upper levels, usually causing the final discourse tree to be very dissimilar to the gold standard. While appropriate post-processing may be employed to fix these errors and help global discourse tree recovery, we feel that it might be more effective to directly improve the raw instance performance of the *Structure* and *Relation* classifiers. Therefore, in our experiments, all classifications are conducted and evaluated on the basis of individual instances.

Each instance is of the form  $(S_L, S_R)$ , which is a

pair of adjacent text spans  $S_L$  (left span) and  $S_R$  (right span), extracted from the discourse tree representation in RST-DT. From each discourse tree, we extract positive instances as those pairs of text spans that are siblings of the same parent node, and negative examples as those pairs of adjacent text spans that are not siblings in the tree structure. In all instances, both  $S_L$  and  $S_R$  must correspond to a constituent in the discourse tree, which can be either an atomic EDU or a concatenation of multiple consecutive EDUs.

### 5.3 Feature extraction

Given a pair of text spans  $(S_L, S_R)$ , we extract the following seven types of features.

**HILDA’s features:** We incorporate the original features used in the HILDA discourse parser with slight modification, which include the following four types of features occurring in  $S_L$ ,  $S_R$ , or both: (1) N-gram prefixes and suffixes; (2) syntactic tag prefixes and suffixes; (3) lexical heads in the constituent parse tree; and (4) POS tag of the dominating nodes.

**Lin et al.’s features:** Following Lin et al. (2009), we extract the following three types of features: (1) pairs of words, one from  $S_L$  and one from  $S_R$ , as originally proposed by Marcu and Echihabi (2002); (2) dependency parse features in  $S_L$ ,  $S_R$ , or both; and (3) syntactic production rules in  $S_L$ ,  $S_R$ , or both.

**Contextual features:** For a globally coherent text, there exist particular sequential patterns in the local usage of different discourse relations. Given  $(S_L, S_R)$ , the pair of text spans of interest, contextual features attempt to encode the discourse relations assigned to the preceding and the following text span pairs. Lin et al. (2009) also incorporated contextual features in their feature set. However, their work was based on PDTB, which has a very different annotation framework from RST-DT (see Section 2): in PDTB, annotated discourse relations can form a chain-like structure such that contextual features can be more readily extracted. However, in RST-DT, a full text is represented as a discourse tree structure, so the *previous* and the *next* discourse relations are not well-defined.

We resolve this problem as follows. Suppose  $S_L = (e_i - e_j)$  and  $S_R = (e_{j+1} - e_k)$ , where  $i \leq j < k$ . To find the previous discourse relation  $REL_{prev}$  that immedi-

ately precedes  $(S_L, S_R)$ , we look for the largest span  $S_{prev} = (e_h - e_{i-1}), h < i$ , such that it ends right before  $S_L$  and all its leaves belong to a single subtree which neither  $S_L$  nor  $S_R$  is a part of. If  $S_L$  and  $S_R$  belong to the same sentence,  $S_{prev}$  must also be a *within-sentence* span, and it must be a *cross-sentence* span if  $S_L$  and  $S_R$  are a cross-sentence span pair.  $REL_{prev}$  is then the discourse relation which covers  $S_{prev}$ . The next discourse relation  $REL_{next}$  that immediately follows  $(S_L, S_R)$  is found in the analogous way.

However, when building a discourse tree using a greedy bottom-up approach, as adopted by the HILDA discourse parser,  $REL_{prev}$  and  $REL_{next}$  are not always available; therefore these contextual features represent an idealized situation. In our experiments we wish to explore whether incorporating perfect contextual features can help better recognize discourse relations, and if so, set an upper bound of performance in more realistic situations.

**Discourse production rules:** Inspired by Lin et al. (2009)’s syntactic production rules as features, we develop another set of production rules, namely discourse production rules, derived directly from the tree structure representation in RST-DT.

For example, with respect to the RST discourse tree shown in Figure 1, we extract the following discourse production rules:  $ATTRIBUTION \rightarrow NO-REL\ NO-REL$ ,  $SAME-UNIT \rightarrow ATTRIBUTION\ NO-REL$ ,  $CONDITION \rightarrow SAME-UNIT\ NO-REL$ , where  $NO-REL$  denotes a leaf node in the discourse subtree.

The intuition behind using discourse production rules is that the discourse tree structure is able to reflect the relatedness of different discourse relations — discourse relations on the lower level of the tree can determine the relation of their direct parent to some degree. Hernault et al. (2010b) attempt to capture such relatedness by traversing a discourse subtree and encoding its traversal path as features, but since they used a depth-first traversal order, the information encoded in a node’s direct children is too distant; whereas most useful information can be gained from the relations covering these direct children.

**Semantic similarities:** Semantic similarities are useful for recognizing relations such as  $COMPARISON$ , when there are no explicit syntactic structures or lexical features signaling such relations.

We use two subsets of similarity features for verbs

and nouns separately. For each verb in either  $S_L$  or  $S_R$ , we look up its most frequent verb class ID in VerbNet<sup>2</sup>, and specify whether that verb class ID appears in  $S_L$ ,  $S_R$ , or both. For nouns, we extract all pairs of nouns from  $(S_L, S_R)$ , and compute the average similarity among these pairs. In particular, we use *path\_similarity*, *lch\_similarity*, *wup\_similarity*, *res\_similarity*, *jcn\_similarity*, and *lin\_similarity* provided in the *nlk.wordnet.similarity* package (Bird et al., 2009) for computing WordNet-based similarity, and always choose the most frequent sense for each noun.

**Cue phrases:** We compile a list of cue phrases, the majority of which are connectives collected by Knott and Dale (1994). For each cue phrase in this list, we determine whether it appears in  $S_L$  or  $S_R$ . If a cue phrase appears in a span, we also determine whether its appearance is in the beginning, the end, or the middle of that span.

## 5.4 Feature selection

If we consider all possible combinations of the features listed in Section 5.3, the resulting data space can be horribly high dimensional and extremely sparse. Therefore, prior to training, we first conduct feature selection to effectively reduce the dimension of the data space.

We employ the same feature selection method as Lin et al. (2009). Feature selection is done for each feature type separately. Among all features belonging to the feature type to be selected, we first extract all possible features that have been seen in the training data, e.g., when applying feature selection for *word pairs*, we find all word pairs that appear in some text span pair that have a discourse relation between them. Then for each extracted feature, we compute its mutual information with all 18 discourse relation classes defined in RST-DT, and use the highest mutual information to evaluate the effectiveness of that feature. All extracted features are sorted to form a ranked list by effectiveness. After that, we use a threshold to select the top features from that ranked list. The total number of selected features used in our experiments is 21,410.

<sup>2</sup><http://verbs.colorado.edu/~mpalmer/projects/verbnet>

## 6 Experiments

As discussed in Section 5.1, our research focus in this paper is the tree-building step of the HILDA discourse parser, which consists of two classifications: *Structure* and *Relation* classification. The binary *Structure* classifier decides whether a discourse relation is likely to hold between consecutive text spans, and the multi-class *Relation* classifier decides which discourse relation label holds between these two text spans if the *Structure* classifier predicts the existence of such a relation.

Although HILDA’s bottom-up approach is aimed at building a discourse tree for the full text, it does not explicitly employ different strategies for *within-sentence* text spans and *cross-sentence* text spans. However, we believe that discourse parsing is significantly more difficult for text spans at higher levels of the discourse tree structure. Therefore, we conduct the following three sub-experiments to explore whether the two classifiers behave differently under different discourse conditions.

**Within-sentence:** Trained and tested on text span pairs belonging to the same sentence.

**Cross-sentence:** Trained and tested on text span pairs belonging to different sentences.

**Hybrid:** Trained and tested on all text span pairs.

In particular, we split the training set and the testing set following the convention of RST-DT, and conduct *Structure* and *Relation* classification by incorporating our rich linguistic features, as listed in Section 5.3 above. To rule out all confounding factors, all classifiers are trained and tested on the basis of individual text span pairs, by assuming the discourse subtree structure (if any) covering each individual text span has been already correctly identified (no error propagation).

### 6.1 Structure classification

The number of training and testing instances used in this experiment for different discourse conditions is listed in Table 1. Instances are extracted in the manner described in Section 5.2. We observe that the distribution of positive and negative instances is extremely skewed for cross-sentence instances, while for all conditions, the distribution is similar in the training and the testing set.

In this experiment, classifiers are trained using

	Dataset	Pos #	Neg #	Total #
<i>Within</i>	Training	11,087	10,188	21,275
	Testing	1,340	1,181	2,521
<i>Cross</i>	Training	6,646	49,467	56,113
	Testing	882	6,357	7,239
<i>Hybrid</i>	Training	17,733	59,655	77,388
	Testing	2,222	7,539	9,761

Table 1: Number of training and testing instances used in *Structure* classification.

the  $SVM^{perf}$  classifier (Joachims, 2005) with a linear kernel.

*Structure* classification performance for all three discourse conditions is shown in Table 2. The columns **Full** and **NC** (No Context) denote the performance of using all features listed in Section 5.3 and all features except for *contextual features* respectively. As discussed in Section 5.3, contextual features represent an ideal situation which is not always available in real applications; therefore, we wish to see how they affect the overall performance by comparing the performance obtained with them and without them as features. The column **HILDA** lists the performance of using Hernault et al. (2010b)’s original features, and **Baseline** denotes the performance obtained by always picking the more frequent class. Performance is measured by four metrics: accuracy, precision, recall, and  $F_1$  score on the test set, shown in the first section in each sub-table.

Under the within-sentence condition, we observe that, surprisingly, incorporating contextual features boosts the overall performance by a large margin, even though it requires only 38 additional features. Under the cross-sentence condition, our features result in lower accuracy and precision than HILDA’s features. However, under this discourse condition, the distribution of positive and negative instances in both training and test sets is extremely skewed, which makes it more sensible to compare the recall and  $F_1$  scores for evaluation. In fact, our features achieve much higher recall and  $F_1$  score despite a much lower precision and a slightly lower accuracy.

In the second section of each sub-table, we also list the  $F_1$  score on the training data. This allows

us to compare the model-fitting capacity of different feature sets from another perspective, especially when the training data is not sufficiently well fitted by the model. For example, looking at the training  $F_1$  score under the cross-sentence condition, we can see that classification using full features and classification without contextual features both perform significantly better on the training data than HILDA does. At the same time, such superior performance is not due to possible over-fitting on the training data, because we are using significantly fewer features (21,410 for **Full** and 21,372 for **NC**) than Hernault et al. (2010b)’s 136,987; rather, it suggests that using carefully selected rich linguistic features is able to better model the problem itself.

Comparing the results obtained under the first two conditions, we see that the binary classification problem of whether a discourse relation is likely to hold between two adjacent text spans is much more difficult under the cross-sentence condition. One major reason is that many features that are predictive for within-sentence instances are no longer applicable (e.g., *Dependency parse features*). In addition, given the extremely imbalanced nature of the dataset under this discourse condition, we might need to employ special approaches to deal with this needle-in-a-haystack problem. This difficulty can also be perceived from the training performance. Compared to the within-sentence condition, all features fit the training data much more poorly under the cross-sentence condition. This suggests that sophisticated features or models in addition to our rich linguistic features must be incorporated in order to fit the problem sufficiently well. Unfortunately, this underfitting issue cannot be resolved by exploiting any abundant linguistic resources for feature vector extension (e.g., Hernault et al. (2010a)), because the poor training performance is no longer caused by the unknown features found in test vectors.

Turning to the hybrid condition, the performance of **Full** features is surprisingly good, probably because we have more available training data than the other two conditions. However, with contextual features removed, our features perform quite similarly to those of Hernault et al. (2010b), but still with a marginal, but nonetheless statistically significant, improvement on recall and  $F_1$  score.

	<b>Full</b>	<b>NC</b>	<b>HILDA</b>	<b>Baseline</b>
<i>Within-sentence</i>				
Accuracy	<b>91.04*</b>	85.17*	83.74	53.15
Precision	<b>92.71*</b>	85.36*	84.81	53.15
Recall	<b>90.22*</b>	87.01*	84.55	100.00
$F_1$	<b>91.45*</b>	86.18*	84.68	69.41
Train $F_1$	<b>97.87*</b>	96.23*	95.42	68.52
<i>Cross-sentence</i>				
Accuracy	87.69	86.68	<b>89.13</b>	87.82
Precision	49.60	44.73	<b>61.90</b>	–
Recall	<b>63.95*</b>	39.46*	28.00	0.00
$F_1$	<b>55.87*</b>	41.93*	38.56	–
Train $F_1$	<b>87.25*</b>	71.93*	49.03	–
<i>Hybrid</i>				
Accuracy	<b>95.64*</b>	87.03	87.04	77.24
Precision	<b>94.77*</b>	74.19	79.41	–
Recall	<b>85.92*</b>	65.98*	58.15	0.00
$F_1$	<b>89.51*</b>	69.84*	67.13	–
Train $F_1$	<b>93.15*</b>	80.79*	72.09	–

Table 2: *Structure* classification performance (in percentage) on text spans of *within-sentence*, *cross-sentence*, and *all* level. Performance that is significantly superior to that of HILDA ( $p < .01$ , using the Wilcoxon sign-rank test for significance) is denoted by \*.

## 6.2 Relation classification

The *Relation* classifier has 18 possible output labels, which are the coarse-grained relation classes defined in RST-DT. We do not consider nuclearity when classifying different discourse relations, i.e., `ATtribution[N][S]` and `ATtribution[S][N]` are treated as the same label. The training and test instances in this experiment are from the positive subset used in *Structure* classification.

In this experiment, classifiers are trained using LibSVM classifier (Chang and Lin, 2011) with a linear kernel and probability estimation.

*Relation* classification performance under three discourse conditions is shown in Table 3. We list the performance achieved by **Full**, **NC**, and **HILDA** features, as well as the majority baseline, which is obtained by always picking the most frequent class label (`ELABORATION` in all cases).

	Full	NC	HILDA	Baseline
<i>Within-sentence</i>				
MAFS	<b>0.490</b>	0.485	0.446	—
WAFS	<b>0.763</b>	0.762	0.740	—
Acc (%)	78.06	<b>78.13</b>	76.42	31.42
TAcc (%)	99.90	<b>99.93</b>	99.26	33.38
<i>Cross-sentence</i>				
MAFS	<b>0.194</b>	0.184	0.127	—
WAFS	<b>0.334</b>	0.329	0.316	—
Acc (%)	<b>46.83</b>	46.71	45.69	42.52
TAcc (%)	<b>78.30</b>	67.30	57.70	47.79
<i>Hybrid</i>				
MAFS	<b>0.440</b>	0.428	0.379	—
WAFS	<b>0.607</b>	0.604	0.588	—
Acc (%)	<b>65.30</b>	65.12	64.18	35.82
TAcc (%)	<b>99.96</b>	99.95	90.11	38.78

Table 3: *Relation* classification performance on text spans of *within-sentence*, *cross-sentence*, and *all* levels.

Following Hernault et al. (2010a), we use Macro-averaged  $F$ -scores (MAFS) to evaluate the performance of each classifier. Macro-averaged  $F$ -score is not influenced by the number of instances that exist in each relation class, by equally weighting the performance of each relation class<sup>3</sup>. Therefore, the evaluation is not biased by the performance on those prevalent classes such as *ATTRIBUTION* and *ELABORATION*. For reasons of space, we do not show the class-wise  $F$ -scores, but in our results, we find that using our features consistently provides superior performance for most class relations over HILDA’s features, and therefore results in higher overall MAFS under all conditions. We also list two other metrics for performance on the test data — Weight-averaged  $F$ -score (WAFS), which weights the performance of each relation class by the number of its existing instances, and the testing accuracy (Acc) — but these metrics are relatively more bi-

<sup>3</sup>No significance test is reported for relation classification, because we are comparing MAFS, which equally weights the performance of each relation. Therefore, traditional significance tests which operate on individual instances rather than individual relation classes are not applicable.

ased evaluation metrics in this task. Similar to *Structure* classification, the accuracy on the training data (TAcc)<sup>4</sup> is listed in the second section of each sub-table. It demonstrates that our carefully selected rich linguistic features are able to better fit the classification problem, especially under the *cross-sentence* condition.

Similar to our observation in *Structure* classification, the performance of *Relation* classification for cross-sentence instances is also much poorer than that on within-sentence instances, which again reveals the difficulty of text-level discourse parsing.

## 7 Conclusions

In this paper, we aimed to develop an RST-style text-level discourse parser. We chose the HILDA discourse parser (Hernault et al., 2010b) as the basis of our work, and significantly improved its tree-building step by incorporating our own rich linguistic features, together with features suggested by Lin et al. (2009). We analyzed the difficulty of extending traditional sentence-level discourse parsing to text-level parsing by showing that using exactly the same set of features, the performance of *Structure* and *Relation* classification on cross-sentence instances is consistently inferior to that on within-sentence instances. We also explored the effect of contextual features on the overall performance. We showed that contextual features are highly effective for both *Structure* and *Relation* classification under all discourse conditions. Although perfect contextual features are available only in idealized situations, when they are correct, together with other features, they can almost correctly predict the tree structure and better predict the relation labels. Therefore, an iterative updating approach, which progressively updates the tree structure and the labeling based on the current estimation, may push the final results toward this idealized end.

Our future work will be to fully implement an end-to-end discourse parser using our rich linguistic features, and focus on improving performance on cross-sentence instances.

<sup>4</sup>We use accuracy instead of MAFS as the evaluation metric on the training data because it is the metric that the training procedure is optimized toward.

## Acknowledgments

This work was financially supported by the Natural Sciences and Engineering Research Council of Canada and by the University of Toronto.

## References

- Jason Baldridge and Alex Lascarides. 2005. Probabilistic head-driven parsing for discourse structure. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 96–103.
- Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python — Analyzing Text with the Natural Language Toolkit*. O’Reilly.
- Lynn Carlson, Daniel Marcu, and Mary Ellen Okunowski. 2001. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In *Proceedings of Second SIGdial Workshop on Discourse and Dialogue*, pages 1–10.
- Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27.
- David A. duVerle and Helmut Prendinger. 2009. A novel discourse parser based on Support Vector Machine classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, Volume 2, ACL ’09*, pages 665–673, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hugo Hernault, Danushka Bollegala, and Mitsuru Ishizuka. 2010a. A semi-supervised approach to improve classification of infrequent discourse relations using feature vector extension. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 399–409, Cambridge, MA, October. Association for Computational Linguistics.
- Hugo Hernault, Helmut Prendinger, David A. duVerle, and Mitsuru Ishizuka. 2010b. HILDA: A discourse parser using support vector machine classification. *Dialogue and Discourse*, 1(3):1–33.
- Thorsten Joachims. 2005. A support vector method for multivariate performance measures. In *International Conference on Machine Learning (ICML)*, pages 377–384.
- Alistair Knott and Robert Dale. 1994. Using linguistic phenomena to motivate a set of coherence relations. *Discourse Processes*, 18(1).
- Huong LeThanh, Geetha Abeyasinghe, and Christian Huyck. 2004. Generating discourse structures for written texts. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 329–335.
- Ziheng Lin, Min-Yen Kan, and Hwee Tou Ng. 2009. Recognizing implicit discourse relations in the Penn Discourse Treebank. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Volume 1, EMNLP ’09*, pages 343–351.
- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2010. A PDTB-styled end-to-end discourse parser. Technical report, School of Computing, National University of Singapore.
- William Mann and Sandra Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281.
- Daniel Marcu and Abdessamad Echihabi. 2002. An unsupervised approach to recognizing discourse relations. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 368–375, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.
- Daniel Marcu. 1997. The rhetorical parsing of natural language texts. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 96–103.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltasakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*.
- David Reitter. 2003. Simple signals for complex rhetorics: On rhetorical analysis with rich-feature support vector models. *LDV Forum*, 18(1/2):38–52.
- Kenji Sagae. 2009. Analysis of discourse structure with syntactic dependencies and data-driven shift-reduce parsing. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 81–84.
- Radu Soricut and Daniel Marcu. 2003. Sentence level discourse parsing using syntactic and lexical information. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, Volume 1*, pages 149–156.
- Rajen Subba and Barbara Di Eugenio. 2009. An effective discourse parser that uses rich linguistic information. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 566–574.
- Bonnie Webber. 2004. D-LTAG: Extending lexicalized TAG to discourse. *Cognitive Science*, 28(5):751–779.



# PDTB-style Discourse Annotation of Chinese Text

**Yuping Zhou**

Computer Science Department  
Brandeis University  
Waltham, MA 02452  
yzhou@brandeis.edu

**Nianwen Xue**

Computer Science Department  
Brandeis University  
Waltham, MA 02452  
xuen@brandeis.edu

## Abstract

We describe a discourse annotation scheme for Chinese and report on the preliminary results. Our scheme, inspired by the Penn Discourse TreeBank (PDTB), adopts the lexically grounded approach; at the same time, it makes adaptations based on the linguistic and statistical characteristics of Chinese text. Annotation results show that these adaptations work well in practice. Our scheme, taken together with other PDTB-style schemes (e.g. for English, Turkish, Hindi, and Czech), affords a broader perspective on how the generalized lexically grounded approach can flesh itself out in the context of cross-linguistic annotation of discourse relations.

## 1 Introduction

In the realm of discourse annotation, the Penn Discourse TreeBank (PDTB) (Prasad et al., 2008) separates itself by adopting a lexically grounded approach: Discourse relations are lexically anchored by discourse connectives (e.g., *because*, *but*, *therefore*), which are viewed as predicates that take abstract objects such as propositions, events and states as their arguments. In the absence of explicit discourse connectives, the PDTB asks the annotator to fill in a discourse connective that best describes the discourse relation between these two sentences, instead of selecting from an inventory of predefined discourse relations. By keeping the discourse annotation lexically grounded even in the case of implicit discourse relations, the PDTB appeals to the annotator’s judgment at an intuitive level. This is in

contrast with an approach in which the set of discourse relations are pre-determined by linguistic experts and the role of the annotator is just to select from those choices (Mann and Thompson, 1988; Carlson et al., 2003). This lexically grounded approach led to consistent and reliable discourse annotation, a feat that is generally hard to achieve for discourse annotation. The PDTB team reported inter-annotator agreement in the lower 90% for explicit discourse relations (Miltasakaki et al., 2004).

In this paper we describe a discourse annotation scheme for Chinese that adopts this lexically grounded approach while making adaptations when warranted by the linguistic and statistical properties of Chinese text. This scheme is shown to be practical and effective in the annotation experiment.

The rest of the paper is organized as follows: In Section 2, we review the key aspects of the PDTB annotation scheme under discussion in this paper. In Section 3, we first show that some key features of Chinese make adaptations necessary in Section 3.1, and then in Section 3.2, we present our systematic adaptations that follow from the differences outlined in Section 3.1. In Section 4, we present the preliminary annotation results we have so far. And finally in Section 5, we conclude the paper.

## 2 The PDTB annotation scheme

As mentioned in the introduction, discourse relation is viewed as a predication with two arguments in the framework of the PDTB. To characterize the predication, the PDTB annotates its argument structure and sense. Two types of discourse relation are distinguished in the annotation: explicit and implicit.

Although their annotation is carried out separately, it conforms to the same paradigm of a discourse connective with two arguments. In what follows, we highlight the key points that will be under discussion in the following sections. To get a more comprehensive and detailed picture of the PDTB scheme, see the PDTB 2.0 annotation manual (Prasad et al., 2007).

## 2.1 Annotation of explicit discourse relations

Explicit discourse relations are those anchored by explicit discourse connectives in text. Explicit connectives are drawn from three grammatical classes:

- **Subordinating conjunctions:** e.g., *because, when, since, although*;
- **Coordinating conjunctions:** e.g., *and, or, nor*;
- **Discourse adverbials:** e.g., *however, otherwise, then, as a result, for example*.

Not all uses of these lexical items are considered to function as a discourse connective. For example, coordinating conjunctions appearing in VP coordinations, such as “*and*” in (1), are not annotated as discourse connectives.

- (1) More common chrysotile fibers are curly and are more easily rejected by the body, Dr. Mossman explained.

The text spans of the two arguments of a discourse connective are marked up. The two arguments, *Arg1* and *Arg2*, are defined based on the physical location of the connective: *Arg2* is the argument expressed by the clause syntactically bound to the connective, and *Arg1* is the other argument. There are no restrictions on how many clauses can be included in the text span for an argument other than the *Minimality Principle*: Only as many clauses and/or sentences should be included in an argument selection as are minimally required and sufficient for the interpretation of the relation.

## 2.2 Annotation of implicit discourse relations

In the case of implicit discourse relations, annotators are asked to insert a discourse connective that best conveys the implicit relation; when no such connective expression is appropriate, the implicit relation is further distinguished as the following three subtypes:

- **AltLex:** when insertion of a connective leads to redundancy due to the presence of an alternatively lexicalized expression, as in (2).
- **EntRel:** when the only relation between the two arguments is that they describe different aspects of the same entity, as in (3).
- **NoRel:** when neither a lexicalized discourse relation nor entity-based coherence is present. It is to be noted that at least some of the “NoRel” cases are due to the *adjacency constraint* (see below for more detail).

- (2) And she further stunned her listeners by revealing her secret garden design method: [*Arg1* Commissioning a friend to spend five or six thousand dollars . . . on books that I ultimately cut up.] [*Arg2* AltLex After that, the layout had been easy.
- (3) [*Arg1* Hale Milgrim, 41 years old, senior vice president, marketing at Elecktra Entertainment Inc., was named president of Capitol Records Inc., a unit of this entertainment concern]. [*Arg2* EntRel Mr. Milgrim succeeds David Berman, who resigned last month].

There are restrictions on what kinds of implicit relations are subjected to annotation, presented below. These restrictions do not have counterparts in explicit relation annotation.

- Implicit relations between adjacent clauses in the same sentence not separated by a semicolon are not annotated, even though the relation may very well be definable. A case in point is presented in (4) below, involving an intra-sentential comma-separated relation between a main clause and a free adjunct.
  - Implicit relations between adjacent sentences across a paragraph boundary are not annotated.
  - The *adjacency constraint*: At least some part of the spans selected for *Arg1* and *Arg2* must belong to the pair of adjacent sentences initially identified for annotation.
- (4) [*MC* The market for export financing was liberalized in the mid-1980s], [*FA* forcing the bank to face competition].

## 2.3 Annotation of senses

Discourse connectives, whether originally present in the data in the case of explicit relations, or filled in by annotators in the case of implicit relations, along with text spans marked as “AltLex”, are annotated with respect to their senses. There are three levels in the sense hierarchy:

- **Class:** There are four major semantic classes: TEMPORAL, CONTINGENCY, COMPARISON, and EXPANSION;
- **Type:** A second level of *types* is further defined for each semantic class. For example, under the class CONTINGENCY, there are two types: “Cause” (relating two situations in a direct cause-effect relation) and “Condition” (relating a hypothetical situation with its (possible) consequences);<sup>1</sup>
- **Subtype:** A third level of *subtypes* is defined for some, but not all, types. For instance, under the type “CONTINGENCY:Cause”, there are two subtypes: “reason” (for cases like *because* and *since*) and “result” (for cases like *so* and *as a result*).

It is worth noting that a type of implicit relation, namely those labeled as “EntRel”, is not part of the sense hierarchy since it has no explicit counterpart.

## 3 Adapted scheme for Chinese

### 3.1 Key characteristics of Chinese text

Despite similarities in discourse features between Chinese and English (Xue, 2005), there are differences that have a significant impact on how discourse relations could be best annotated. These differences can be illustrated with (5):

- (5) 据悉 , [AO1 东莞 海关  
according to reports , Dongguan Customs  
共 接受 企业 合同 备案  
in total accept company contract record  
八千四百多 份] , [AO2 比 试点  
8400 plus CLASS , compare pilot  
前 略 有 上升] , [AO3 企业  
before slight EXIST increase , company

<sup>1</sup>There is another dimension to this level, i.e. literal or pragmatic use. If this dimension is taken into account, there could be said to be *four* types: “Cause”, “Pragmatic Cause”, “Condition”, and “Pragmatic Condition”. For details, see Prasad et al. (2007).

反应 良好] , [AO4 普遍  
respond/response well/good , generally  
表示 接受] 。  
acknowledge accept/acceptance .

“According to reports, [AO1 Dongguan District Customs accepted more than 8400 records of company contracts], [AO2 a slight increase from before the pilot]. [AO3 Companies responded well], [AO4 generally acknowledging acceptance].”

This sentence reports on how a pilot program worked in Dongguan City. Because all that is said is about the pilot program, it is perfectly natural to include it all in a single sentence in Chinese. Intuitively though, there are two different aspects of how the pilot program worked: the number of records and the response from the affected companies. To report the same facts in English, it is more natural to break them down into two sentences or two semi-colon-separated clauses, but in Chinese, *not only are they merely separated by comma, but also there is no connective relating them.*

This difference in writing style necessitates rethinking of the annotation scheme. If we apply the PDTB scheme to the English translation, regardless of whether the two pieces of facts are expressed in two sentences or two semi-colon-separated clauses, at least one discourse relation will be annotated, relating these two text units. In contrast, if we apply the same scheme to the Chinese sentence, no discourse relation will be picked out because this is just one comma-separated sentence with no explicit discourse connectives in it. In other words, the discourse relation within the Chinese sentence, which would be captured in its English counterpart following the PDTB procedure, would be lost when annotating Chinese. Such loss is not a sporadic occurrence but rather a very prevalent one since it is associated with the customary writing style of Chinese. To ensure a reasonable level of coverage, we need to consider comma-delimited intra-sentential implicit relations when annotating Chinese text.

There are some complications associated with this move. One of them is that it introduces into discourse annotation considerable ambiguity associated with the comma. For example, the first instance of comma in (5), immediately following “据悉” (“according to reports”), clearly does not indicate a discourse relation, so it needs to be spelt out in

the guidelines how to exclude such cases of comma as discourse relation indicators. We think, however, that disambiguating the commas in Chinese text is valuable in its own right and is a necessary step in annotating discourse relations.

Another complication is that some comma-separated chunks are ambiguous as to whether they should be considered potential arguments in a discourse relation. The chunks marked AO2 and AO4 in (5) are examples of such cases. They, judging from their English translation, may seem clear cases of *free adjuncts* in PDTB terms (Prasad et al., 2007), but there is no justification for treating them as such in Chinese. The lack of justification comes from at least three features of Chinese:

- Certain words, for instance, “反应” (“respond/response”), “良好” (“well/good”) and “接受” (“accept/acceptance”), are ambiguous with respect to their POS, and when they combine, the resulting sentence may have more than one syntactic analysis. For example, AO3 may be literally translated as “Companies responded well” or “Companies’ response was good”.
- There are no inflectional clues to differentiate free adjuncts and main clauses. For example, one can be reasonably certain that “表示” (“acknowledge”) functions as a verb in (5), however, there is no indication whether it is in the form corresponding to “acknowledging” or “acknowledged” in English. Or putting it differently, whether one wants to express in Chinese the meaning corresponding to the *-ing* form or the tensed form in English, the same form “表示” could apply.
- Both subject and object can be dropped in Chinese, and they often are when they are inferable from the context. For example, in the two-sentence sequence below, the subject of (7) is dropped since it is clearly the same as the subject of the previous sentence in (6).

(6) [S<sub>1</sub> 近五年来, 上海  
recent five years since, Shanghai through  
通过积极从外省市  
actively from other province city procure  
收购出口货源、举办中国  
export supply, organize China East

华东出口商品交易会等  
Export Commodity Fair etc. event,  
活动, 增强口岸对 全国  
strengthen port to whole country DE  
的辐射能力。]  
connection capability .

“[S<sub>1</sub> In the past five years, Shanghai strengthened the connection of its port to other areas of the country through actively procuring export supplies from other provinces and cities, and through organizing events such as the East China Export Commodities Fair.]”

(7) [S<sub>2</sub> 同时, 发展  
At the same time, develop  
跨国经营, 大力开拓  
transnational operation, vigorously open up  
多元化市场。]  
diversified market

“[S<sub>2</sub> At the same time, (it) developed transnational operations (and) vigorously opened up diversified markets.]”

Since the subject can be omitted from the entire sentence, absence or presence of subject in a clause is not an indication whether the clause is a main clause or a free adjunct, or whether it is part of a VP coordination without a connective. So if we take into account both the lack of differentiating inflectional clues and the possibility of omitting the subject, AO4 in (5) may be literally translated as “generally acknowledging acceptance”, or “(and) generally acknowledged acceptance”, or “(companies) generally acknowledged acceptance”, or “(companies) generally acknowledged (they) accepted (it)”.

Since in Chinese, there is no reliable indicator distinguishing between main clauses and free adjuncts, or distinguishing between coordination on the clause level without the subject and coordination on the VP level, *we will not rely on these distinctions in annotation*, as the PDTB team does in their annotation.

These basic decisions directly based on linguistic characteristics of Chinese lead to more systematic adaptations to the annotation scheme, to which we will turn in the next subsection.

### 3.2 Systematic adaptations

The main consequence of the basic decisions described in Section 3.1 is that we have a whole lot

more tokens of implicit relation than explicit relation to deal with. According to a rough count on 20 randomly selected files from Chinese Treebank (Xue et al., 2005), 82% are tokens of implicit relation, compared to 54.5% in the PDTB 2.0. Given the overwhelming number of implicit relations, we re-examine where it could make an impact in the annotation scheme. There are three such areas.

### 3.2.1 Procedural division between explicit and implicit discourse relation

In the PDTB, explicit and implicit relations are annotated separately. This is probably partly because explicit connectives are quite abundant in English, and partly because the project evolved in stages, expanding from the more canonical case of explicit relation to implicit relation for greater coverage. When annotating Chinese text, maintaining this procedural division makes much less sense: the landscape of discourse relation (or at least the key elements of it) has already been mapped out by the PDTB work and to set up a separate task to cover 18% of the data does not seem like a worthwhile bother without additional benefits for doing so.

So the question now is *how to annotate explicit and implicit relations in one fell swoop?* In Chinese text, the use of a discourse connective is almost always accompanied by a punctuation or two (usually period and/or comma), preceding or flanking it. So a sensible solution is to rely on punctuations as the denominator between explicit and implicit relations; and in the case of explicit relation, the connective will be marked up as an attribute of the discourse relation. This unified approach simplifies the annotation procedure while preserving the explicit/implicit distinction in the process.

One might question, at this point, whether such an approach can still call itself “lexically grounded”. Certainly not if one interprets the term literally; but in a broader sense, our approach can be seen as an instantiation of a generalized version of it, much the same way that the PDTB is an, albeit different, instantiation of it for English. The thrust of the lexically grounded approach is that discourse annotation should be a data-driven, bottom-up process, rather than a top-down one, trying to fit data into a prescriptive system. Once the insight that a discourse connective functions like a predicate with two ar-

guments is generalized to cover *all* discourse relations, there is no fundamental difference between explicit and implicit discourse relations: both work like a predicate whether or not there is a lexicalization of it. As to what role this distinction plays in the annotation procedure, it is an engineering issue, depending on a slew of factors, among which are cross-linguistic variations. In the case of Chinese, we think it is more economical to treat explicit and implicit relations alike in the annotation process.

To treat explicit and implicit relations alike actually goes beyond annotating them in one pass; it also involves how they are annotated, which we discuss next.

### 3.2.2 Annotation of implicit discourse relations

In the PDTB, treatment of implicit discourse relations is modeled after that of explicit relations, and at the same time, some restrictions are put on implicit, but not explicit, relations. This is quite understandable: implicit discourse relations tend to be vague and elusive, so making use of explicit relations as a prototype helps pin them down, and restrictions are put in place to strike a balance between high reliability and good coverage. When implicit relations constitute a vast majority of the data as is the case with Chinese, both aspects need to be re-examined to strike a new balance.

In the PDTB, annotators are asked to insert a discourse connective that best conveys the implicit discourse relation between two adjacent discourse units; when no such connective expression is appropriate, the implicit discourse relation is further distinguished as “AltLex”, “EntRel”, and “NoRel”. The inserted connectives and those marked as “AltLex”, along with explicit discourse connectives, are further annotated with respect to their senses.

When a connective needs to be inserted in a majority of cases, the difficulty of the task really stands out. In many cases, it seems, there is a good reason for not having a connective present and because of it, the wording rejects insertion of a connective even if it expresses the underlying discourse relation exactly (or sometimes, maybe the wording itself is the reason for not having a connective). So to try to insert a connective expression may very well be too hard a task for annotators, with little to show for their effort in the end.

Furthermore, the inter-annotator agreement for providing an explicit connective in place of an implicit one is computed based on the *type* of explicit connectives (e.g. cause-effect relations, temporal relations, contrastive relations, etc.), rather than based on their identity (Miltsakaki et al., 2004). This suggests that a reasonable degree of agreement for such a task may only be reached with a coarse classification scheme.

Given the above two considerations, our solution is to annotate implicit discourse relations with their senses directly, bypassing the step of inserting a connective expression. It has been pointed out that to train annotators to reason about pre-defined abstract relations with high reliability might be too hard a task (Prasad et al., 2007). This difficulty can be overcome by associating each semantic type with one or two prototypical explicit connectives and asking annotators to consider each to see if it expresses the implicit discourse relation. This way, annotators have a concrete aid to reason about abstract relations without having to choose one connective from a set expressing roughly the same relation or having to worry about whether insertion of the connective is somehow awkward.

It should be noted that annotating implicit relations directly with their senses means that sense annotation is no longer restricted to those that can be lexically expressed, but also includes those that cannot, notably those labeled “EntRel/NoRel” in the PDTB.<sup>2</sup> In other words, we annotate senses of *discourse relations*, not just connectives and their lexical alternatives (in the case of AltLex). This expansion is consistent with the generalized view of the lexically grounded approach discussed in Section 3.2.1.

With respect to restrictions on implicit relation, we will adopt them as they prove to be necessary in the annotation process, with one exception. The exception is the restriction that implicit relations between adjacent clauses in the same sentence not separated by a semi-colon are not annotated. This restriction seems to apply mainly to a main clause and any free adjunct attached to it in English; in Chinese, however, the distinction between a main clause and a

free adjunct is not as clear-cut for reasons explained in Section 3.1. So this restriction is not applicable for Chinese annotation.

### 3.2.3 Definition of Arg1 and Arg2

The third area that an overwhelming number of implicit relation in the data affects is how *Arg1* and *Arg2* are defined. As mentioned in the introduction, discourse relations are viewed as a predication with two arguments. These two arguments are defined based on the physical location of the connective in the PDTB: *Arg2* is the argument expressed by the clause syntactically bound to the connective and *Arg1* is the other argument. In the case of implicit relations, the label is assigned according to the text order.

In an annotation task where implicit relations constitute an overwhelming majority, the distinction of *Arg1* and *Arg2* is meaningless in most cases. In addition, the phenomenon of parallel connectives is predominant in Chinese. Parallel connectives are pairs of connectives that take the same arguments, examples of which in English are “*if..then*”, “*either..or*”, and “*on the one hand..on the other hand*”. In Chinese, most connectives are part of a pair; though some can be dropped from their pair, it is considered “proper” or formal to use both. (8) below presents two such examples, for which parallel connectives are not possible in English.

- (8) a. 伦敦 股市 因 适逢  
London stock market because coincide  
银行节 , 故 没有 开市。  
Bank Holiday , therefore NEG open market  
“London Stock Market did not open because it was Bank Holiday.”
- b. 虽然 他们 不 离 土 、不 离  
Although they NEG leave land , NEG leave  
乡 , 但 严格 来 讲 已  
home village , but strict PART speak already  
不再 是 传统 意义 上 的 农民。  
no longer be tradition sense PREP DE peasant  
“Although they do not leave land or their home village, strictly speaking, they are no longer peasants in the traditional sense.”

In the PDTB, parallel connectives are annotated discontinuously; but given the prevalence of such phenomenon in Chinese, such practice would generate

<sup>2</sup>Thus “EntRel” and “NoRel” are treated as relation senses, rather than relation types, in our scheme.

a considerably high percentage of essentially repetitive annotation among explicit relations.

So the situation with Chinese is that distinguishing *Arg1* and *Arg2* the PDTB way is meaningless in most cases, and in the remaining cases, it often results in duplication. Rather than abandoning the distinction altogether, we think it makes more sense to define *Arg1* and *Arg2* semantically. It will not create too much additional work beyond distinction of different senses of discourse relation in the PDTB. For example, in the semantic type CONTINGENCY:Cause, we can define “reason” as *Arg1* and “result” as *Arg2*. In this scheme, no matter which one of 因 (“because”) and 故 (“therefore”) appears without the other, or if they appear as a pair in a sentence, or if the relation is implicit, the *Arg1* and *Arg2* labels will be consistently assigned to the same clauses.

This approach is consistent with the move from annotating senses of *connectives* to annotating senses of *discourse relations*, pointed out in Section 3.2.2. For example, in the PDTB’s sense hierarchy, “reason” and “result” are subtypes under type CONTINGENCY:Cause: “reason” applies to connectives like “because” and “since” while “result” applies to connectives like “so” and “as a result”. When we move to annotating senses of discourse relations, since both types of connectives express the same underlying discourse relation, there will not be further division under CONTINGENCY:Cause, and the “reason”/“result” distinction is an intrinsic property of the semantic type. We think this level of generality makes sense semantically.

## 4 Annotation experiment

To test our adapted annotation scheme, we have conducted annotation experiments on a modest, yet significant, amount of data and computed agreement statistics.

### 4.1 Set-up

The agreement statistics come from annotation conducted by two annotators in training so far. The data set consists of 98 files taken from the Chinese Treebank (Xue et al., 2005). The source of these files is Xinhua newswire. The annotation is carried out on

the PDTB annotation tool<sup>3</sup>.

### 4.2 Inter-annotator agreement

To evaluate our proposed scheme, we measure agreement on each adaption proposed in Section 3, as well as agreement on argument span determination. Whenever applicable, we also present (roughly) comparable statistics of the PDTB (Mitsakaki et al., 2004). The results are summarized in Table 1.

	Chinese		PDTB (%)
	tkn no.	F(p/r) (%)	
<i>rel-ident</i>	3951*	95.4 (96.0/94.7)	N/A
<i>rel-type</i>	3951	95.1	N/A
<i>imp-sns-type</i>	2967	87.4	72
<i>arg-order</i>	3059	99.8	N/A
<b>argument span</b>			
<i>exp-span-xm</i>	1580	84.2	90.2
<i>exp-span-pm</i>	1580	99.6	94.5
<i>imp-span-xm</i>	5934	76.9	85.1
<i>overall-bnd-</i>	14039*	87.7 (87.5/87.9)	N/A

Table 1: Inter-annotator agreement in various aspects of Chinese discourse annotation: *rel-ident*, discourse relation identification; *rel-type*, relation type classification; *imp-sns-type*, classification of sense type of implicit relations; *arg-order*, order determination of *Arg1* and *Arg2*. For agreement on argument spans, the naming convention is <type-of-relation>-<element-as-independent-token>-<matching-method>. *exp*: explicit relations; *imp*: implicit relations; *span*: argument span; *xm*: exact match; *pm*: partial match; *bnd*: boundary. \*: number of tokens agreed on by both annotators.

The first adaption we proposed is to annotate explicit and implicit discourse relations in one pass. This introduces two steps, at which agreement can each be measured: First, the annotator needs to make the judgment, at each instance of the punctuations, whether there is a discourse relation (a step we call “relation identification”); second, once a discourse relation is identified, the annotator needs to classify the type as one of “Explicit”, “Implicit”, or “AltLex” (a step we call “relation type classification”). The agreement at these two steps is 95.4%

<sup>3</sup><http://www.seas.upenn.edu/~pdtb/tools.shtml#annotator>

and 95.1% respectively.

The second adaption is to bypass the step of inserting a connective when annotating an implicit discourse relation and classify the sense directly. The third adaptation is to define Arg1 and Arg2 semantically for each sense. To help annotators think about relation sense abstractly and determine the order of the arguments, we put a helper item alongside each sense label, like “Causation: 因为arg1所以arg2” (“Causation: *because arg1 therefore arg2*”). This approach works well, as evidenced by 87.4%<sup>4</sup> and 99.8% agreement for the two processes respectively.

To evaluate agreement on determining argument span, we adopt four measures. In the first three, explicit and implicit relations are calculated separately (although they are actually annotated in the same process) to make our results comparable to the published PDTB results. Each argument span is treated as an independent token and either exact or partial match (i.e. if two spans share one boundary) counts as 1. The fourth measure is less stringent than exact match and more stringent than partial match: It groups explicit and implicit relation together and treats each boundary as an independent token. Typically, an argument span has two boundaries, but it can have four (or more) boundaries when an argument span is interrupted by a connective and/or an AltLex item.

Evidently, determining argument span is the most challenging aspect of discourse annotation. However, it should be pointed out that agreement was on an overall upward trend, which became especially prominent after we instituted a restriction on implicit relations across a paragraph boundary towards the end of the training period. It restricts full anno-

<sup>4</sup>Two more points should be made about this number. First, it may be partially attributed to our differently structured sense hierarchy. It is a flat structure containing the following 12 values: ALTERNATIVE, CAUSATION, CONDITIONAL, CONJUNCTION, CONTRAST, EXPANSION, PROGRESSION, PURPOSE, RESTATEMENT, TEMPORAL, EntRel, and NoRel. Aside from including EntRel and NoRel (the reason and significance of which have been discussed in Section 3.2.2), the revision was by and large not motivated by Chinese-specific features, so we do not address it in detail in this paper. Second, in making the comparison with the PDTB result, the 12-value structure is collapsed into 5 values: TEMPORAL, CONTINGENCY, COMPARISON, EXPANSION, and EntRel/NoRel, which must be different from the 5 values in Miltsakaki et al. (2004), judging from the descriptions.

tation to only three specific situations so that most loose and/or hard-to-delimit relations across paragraph boundaries are excluded. This restriction appears to be quite effective, as shown in Table 2.

	num of rel.'s	Overall Arg Span	
		boundary F(p/r) (%)	span-em (%)
last 5 wks	1103	90.0 (90.0/89.9)	80.8
last 3 wks	677	91.0 (91.0/91.0)	82.5
last 2 wks	499	91.8 (91.8/91.8)	84.2

Table 2: Inter-annotator agreement on argument span during the last 5 weeks of training.

## 5 Conclusions

We have presented a discourse annotation scheme for Chinese that adopts the lexically ground approach of the PDTB while making systematic adaptations motivated by characteristics of Chinese text. These adaptations not only work well in practice, as evidenced by the results from our annotation experiment, but also embody a more generalized view of the lexically ground approach to discourse annotation: Discourse relations are predication involving two arguments; the predicate can be either covert (i.e. Implicit) or overt, lexicalized as discourse connectives (i.e. Explicit) or their more polymorphous counterparts (i.e. AltLex). Consistent with this generalized view is a more semantically motivated sense annotation scheme: Senses of *discourse relations* (as opposed to just connectives) are annotated; and the two arguments of the discourse relation are semantically defined, allowing the sense structure to be more general and less connective-dependent. These framework-level generalizations can be applied to discourse annotation of other languages.

## Acknowledgments

This work is supported by the IIS Division of the National Science Foundation via Grant No. 0910532 entitled “Richer Representations for Machine Translation” and by the CNS Division via Grant No. 0855184 entitled “Building a community resource for temporal inference in Chinese”. All views expressed in this paper are those of the authors and do



not necessarily represent the view of the National Science Foundation.

## References

- Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a Discourse-Tagged Corpus in the Framework of Rhetorical Structure Theory. In *Current Directions in Discourse and Dialogue*. Kluwer Academic Publishers.
- William Mann and Sandra Thompson. 1988. Rhetorical structure theory. Toward a functional theory of text organization. *Text*, 8(3):243–281.
- Eleni Miltsakaki, Rashmi Prasad, Aravind Joshi, and Bonnie Webber. 2004. Annotating discourse connectives and their arguments. In *Proceedings of the HLT/NAACL Workshop on Frontiers in Corpus Annotation*, pages 9–16, Boston, MA, May.
- Rashmi Prasad, Eleni Miltsakaki, Nikhil Dinesh, Alan Lee, Aravind Joshi, Livio Robaldo, and Bonnie Webber, 2007. *The Penn Discourse Treebank 2.0 Annotation Manual*. The PDTB Research Group, December.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. The Penn Discourse Treebank 2.0. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*.
- Nianwen Xue, Fei Xia, Fu dong Chiou, and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207–238.
- Nianwen Xue. 2005. Annotating the Discourse Connectives in the Chinese Treebank. In *Proceedings of the ACL Workshop on Frontiers in Corpus Annotation*, Ann Arbor, Michigan.

# SITS: A Hierarchical Nonparametric Model using Speaker Identity for Topic Segmentation in Multiparty Conversations

**Viet-An Nguyen**  
Department of Computer Science  
and UMIACS  
University of Maryland  
College Park, MD  
vietan@cs.umd.edu

**Jordan Boyd-Graber**  
iSchool  
and UMIACS  
University of Maryland  
College Park, MD  
jbg@umiacs.umd.edu

**Philip Resnik**  
Department of Linguistics  
and UMIACS  
University of Maryland  
College Park, MD  
resnik@umd.edu

## Abstract

One of the key tasks for analyzing conversational data is segmenting it into coherent topic segments. However, most models of topic segmentation ignore the *social* aspect of conversations, focusing only on the words used. We introduce a hierarchical Bayesian nonparametric model, Speaker Identity for Topic Segmentation (SITS), that discovers (1) the topics used in a conversation, (2) how these topics are shared across conversations, (3) when these topics shift, and (4) a person-specific tendency to introduce new topics. We evaluate against current unsupervised segmentation models to show that including person-specific information improves segmentation performance on meeting corpora and on political debates. Moreover, we provide evidence that SITS captures an individual's tendency to introduce new topics in political contexts, via analysis of the 2008 US presidential debates and the television program Crossfire.

## 1 Topic Segmentation as a Social Process

Conversation, interactive discussion between two or more people, is one of the most essential and common forms of communication. Whether in an informal situation or in more formal settings such as a political debate or business meeting, a conversation is often not about just one thing: topics evolve and are replaced as the conversation unfolds. Discovering this hidden structure in conversations is a key problem for conversational assistants (Tur et al., 2010) and tools that summarize (Murray et al., 2005) and display (Ehlen et al., 2007) conversational data. Topic segmentation also can illuminate individuals' agendas (Boydston et al., 2011), patterns of agreement and disagreement (Hawes et al., 2009; Abbott

et al., 2011), and relationships among conversational participants (Ireland et al., 2011).

One of the most natural ways to capture conversational structure is topic segmentation (Reynar, 1998; Purver, 2011). Topic segmentation approaches range from simple heuristic methods based on lexical similarity (Morris and Hirst, 1991; Hearst, 1997) to more intricate generative models and supervised methods (Georgescu et al., 2006; Purver et al., 2006; Gruber et al., 2007; Eisenstein and Barzilay, 2008), which have been shown to outperform the established heuristics.

However, previous computational work on conversational structure, particularly in topic discovery and topic segmentation, focuses primarily on content, ignoring the speakers. We argue that, because conversation is a *social* process, we can understand conversational phenomena better by explicitly modeling behaviors of conversational participants. In Section 2, we incorporate participant identity in a new model we call Speaker Identity for Topic Segmentation (SITS), which discovers topical structure in conversation while jointly incorporating a participant-level social component. Specifically, we explicitly model an individual's tendency to introduce a topic. After outlining inference in Section 3 and introducing data in Section 4, we use SITS to improve state-of-the-art-topic segmentation and topic identification models in Section 5. In addition, in Section 6, we also show that the per-speaker model is able to discover individuals who shape and influence the course of a conversation. Finally, we discuss related work and conclude the paper in Section 7.

## 2 Modeling Multiparty Discussions

**Data Properties** We are interested in turn-taking, multiparty discussion. This is a broad category, in-

cluding political debates, business meetings, and on-line chats. More formally, such datasets contain  $C$  conversations. A conversation  $c$  has  $T_c$  turns, each of which is a maximal uninterrupted utterance by one speaker.<sup>1</sup> In each turn  $t \in [1, T_c]$ , a speaker  $a_{c,t}$  utters  $N$  words  $\{w_{c,t,n}\}$ . Each word is from a vocabulary of size  $V$ , and there are  $M$  distinct speakers.

**Modeling Approaches** The key insight of topic segmentation is that segments evince lexical cohesion (Galley et al., 2003; Olney and Cai, 2005). Words within a segment will look more like their neighbors than other words. This insight has been used to tune supervised methods (Hsueh et al., 2006) and inspire unsupervised models of lexical cohesion using bags of words (Purver et al., 2006) and language models (Eisenstein and Barzilay, 2008).

We too take the unsupervised statistical approach. It requires few resources and is applicable in many domains without extensive training. Like previous approaches, we consider each turn to be a bag of words generated from an admixture of topics. Topics—after the topic modeling literature (Blei and Lafferty, 2009)—are multinomial distributions over terms. These topics are part of a generative model posited to have produced a corpus.

However, topic models alone cannot model the dynamics of a conversation. Topic models typically do not model the temporal dynamics of individual documents, and those that do (Wang et al., 2008; Gerrish and Blei, 2010) are designed for larger documents and are not applicable here because they assume that most topics appear in every time slice.

Instead, we endow each turn with a binary latent variable  $l_{c,t}$ , called the *topic shift*. This latent variable signifies whether the speaker changed the topic of the conversation. To capture the topic-controlling behavior of the speakers across different conversations, we further associate each speaker  $m$  with a latent *topic shift tendency*,  $\pi_m$ . Informally, this variable is intended to capture the propensity of a speaker to effect a topic shift. Formally, it represents the probability that the speaker  $m$  will change the topic (distribution) of a conversation.

We take a Bayesian nonparametric approach (Müller and Quintana, 2004). Unlike

<sup>1</sup>Note the distinction with phonetic utterances, which by definition are bounded by silence.

parametric models, which *a priori* fix the number of topics, nonparametric models use a flexible number of topics to better represent data. Nonparametric distributions such as the Dirichlet process (Ferguson, 1973) share statistical strength among conversations using a hierarchical model, such as the hierarchical Dirichlet process (HDP) (Teh et al., 2006).

## 2.1 Generative Process

In this section, we develop SITS, a generative model of multiparty discourse that jointly discovers topics and speaker-specific topic shifts from an unannotated corpus (Figure 1a). As in the hierarchical Dirichlet process (Teh et al., 2006), we allow an unbounded number of topics to be shared among the turns of the corpus. Topics are drawn from a base distribution  $H$  over multinomial distributions over the vocabulary, a finite Dirichlet with symmetric prior  $\lambda$ . Unlike the HDP, where every document (here, every turn) draws a new multinomial distribution from a Dirichlet process, the social and temporal dynamics of a conversation, as specified by the binary topic shift indicator  $l_{c,t}$ , determine when new draws happen.

The full generative process is as follows:

1. For speaker  $m \in [1, M]$ , draw speaker shift probability  $\pi_m \sim \text{Beta}(\gamma)$
2. Draw global probability measure  $G_0 \sim \text{DP}(\alpha, H)$
3. For each conversation  $c \in [1, C]$ 
  - (a) Draw conversation distribution  $G_c \sim \text{DP}(\alpha_0, G_0)$
  - (b) For each turn  $t \in [1, T_c]$  with speaker  $a_{c,t}$ 
    - i. If  $t = 1$ , set the topic shift  $l_{c,t} = 1$ . Otherwise, draw  $l_{c,t} \sim \text{Bernoulli}(\pi_{a_{c,t}})$ .
    - ii. If  $l_{c,t} = 1$ , draw  $G_{c,t} \sim \text{DP}(\alpha_c, G_c)$ . Otherwise, set  $G_{c,t} \equiv G_{c,t-1}$ .
    - iii. For each word index  $n \in [1, N_{c,t}]$ 
      - Draw  $\psi_{c,t,n} \sim G_{c,t}$
      - Draw  $w_{c,t,n} \sim \text{Multinomial}(\psi_{c,t,n})$

The hierarchy of Dirichlet processes allows statistical strength to be shared across contexts; within a conversation and across conversations. The per-speaker topic shift tendency  $\pi_m$  allows speaker *identity* to influence the evolution of topics.

To make notation concrete and aligned with the topic segmentation, we introduce notation for *segments* in a conversation. A segment  $s$  of conversation  $c$  is a sequence of turns  $[\tau, \tau']$  such that  $l_{c,\tau} = l_{c,\tau'+1} = 1$  and  $l_{c,t} = 0, \forall t \in (\tau, \tau']$ . When  $l_{c,t} = 0$ ,  $G_{c,t}$  is the same as  $G_{c,t-1}$  and all topics (i.e. multinomial distributions over words)  $\{\psi_{c,t,n}\}$  that generate words in turn  $t$  and the topics  $\{\psi_{c,t-1,n}\}$  that generate words in turn  $t - 1$  come from the same

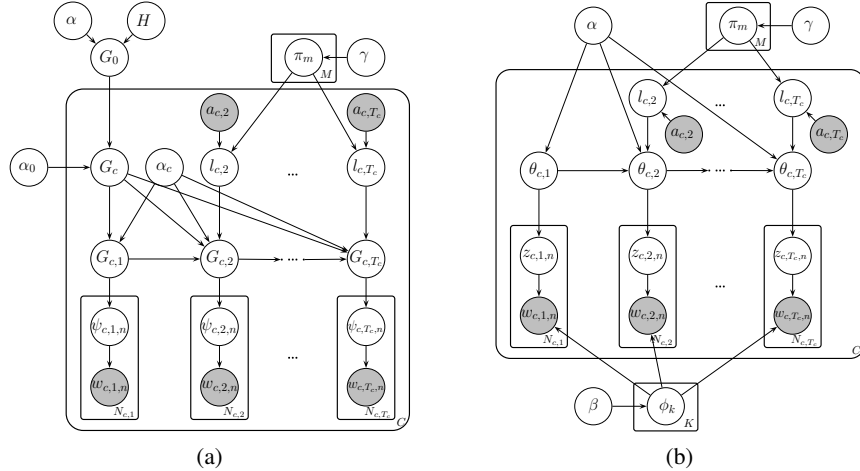


Figure 1: Graphical model representations of our proposed models: (a) the nonparametric version; (b) the parametric version. Nodes represent random variables (shaded ones are observed), lines are probabilistic dependencies. Plates represent repetition. The innermost plates are turns, grouped in conversations.

distribution. Thus all topics used in a segment  $s$  are drawn from a single distribution,  $G_{C,s}$ ,

$$G_{c,s} \mid l_{c,1}, l_{c,2}, \dots, l_{c,T_c}, \alpha_c, G_c \sim \text{DP}(\alpha_c, G_c) \quad (1)$$

For notational convenience,  $S_c$  denotes the number of segments in conversation  $c$ , and  $s_t$  denotes the segment index of turn  $t$ . We emphasize that all segment-related notations are derived from the posterior over the topic shifts  $l$  and not part of the model itself.

**Parametric Version** SITS is a generalization of a parametric model (Figure 1b) where each turn has a multinomial distribution over  $K$  topics. In the parametric case, the number of topics  $K$  is fixed. Each topic, as before, is a multinomial distribution  $\phi_1 \dots \phi_K$ . In the parametric case, each turn  $t$  in conversation  $c$  has an explicit multinomial distribution over  $K$  topics  $\theta_{c,t}$ , identical for turns within a segment. A new topic distribution  $\theta$  is drawn from a Dirichlet distribution parameterized by  $\alpha$  when the topic shift indicator  $l$  is 1.

The parametric version does not share strength within or across conversations, unlike SITS. When applied on a single conversation without speaker identity (all speakers are identical) it is equivalent to (Purver et al., 2006). In our experiments (Section 5), we compare against both.

### 3 Inference

To find the latent variables that best explain observed data, we use Gibbs sampling, a widely used Markov chain Monte Carlo inference technique (Neal, 2000; Resnik and Hardisty, 2010). The state space is latent variables for topic indices assigned to all tokens  $\mathbf{z} = \{z_{c,t,n}\}$  and topic shifts assigned to turns  $\mathbf{l} = \{l_{c,t}\}$ . We marginalize over all other latent variables. Here, we only present the conditional sampling equations; for more details, see our supplement.<sup>2</sup>

#### 3.1 Sampling Topic Assignments

To sample  $z_{c,t,n}$ , the index of the shared topic assigned to token  $n$  of turn  $t$  in conversation  $c$ , we need to sample the *path* assigning each word token to a segment-specific topic, each segment-specific topic to a conversational topic and each conversational topic to a shared topic. For efficiency, we make use of the *minimal path assumption* (Wallach, 2008) to generate these assignments.<sup>3</sup> Under the minimal path assumption, an observation is assumed to have been generated by using a new distribution if and only if there is no existing distribution with the same value.

<sup>2</sup> <http://www.cs.umd.edu/~vietan/topicshift/appendix.pdf>

<sup>3</sup> We also investigated using the maximal assumption and fully sampling assignments. We found the minimal path assumption worked as well as explicitly sampling seating assignments and that the maximal path assumption worked less well.

We use  $N_{c,s,k}$  to denote the number of tokens in segment  $s$  in conversation  $c$  assigned topic  $k$ ;  $N_{c,k}$  denotes the total number of segment-specific topics in conversation  $c$  assigned topic  $k$  and  $N_k$  denotes the number of conversational topics assigned topic  $k$ .  $TW_{k,w}$  denotes the number of times the shared topic  $k$  is assigned to word  $w$  in the vocabulary. Marginal counts are represented with  $\cdot$  and  $*$  represents all hyperparameters. The conditional distribution for  $z_{c,t,n}$  is  $P(z_{c,t,n} = k \mid w_{c,t,n} = w, \mathbf{z}^{-c,t,n}, \mathbf{w}^{-c,t,n}, \mathbf{l}, *) \propto$

$$\frac{N_{c,s_t,k}^{-c,t,n} + \alpha_c \frac{N_{c,k}^{-c,t,n} + \alpha_0 \frac{N_{c,t,n}^{-c,t,n} + \frac{\alpha_c}{K}}{N_{c,t,n}^{-c,t,n} + \alpha_0}}{N_{c,s_t,\cdot}^{-c,t,n} + \alpha_c} \times \begin{cases} \frac{TW_{k,w}^{-c,t,n} + \lambda}{TW_{k,\cdot}^{-c,t,n} + V\lambda}, \\ \frac{1}{V} & k \text{ new.} \end{cases} \quad (2)$$

Here  $V$  is the size of the vocabulary,  $K$  is the current number of shared topics and the superscript  $^{-c,t,n}$  denotes counts without considering  $w_{c,t,n}$ . In Equation 2, the first factor is proportional to the probability of sampling a path according to the minimal path assumption; the second factor is proportional to the likelihood of observing  $w$  given the sampled topic. Since an uninformed prior is used, when a new topic is sampled, all tokens are equiprobable.

### 3.2 Sampling Topic Shifts

Sampling the topic shift variable  $l_{c,t}$  requires us to consider merging or splitting segments. We use  $\mathbf{k}_{c,t}$  to denote the shared topic indices of all tokens in turn  $t$  of conversation  $c$ ;  $S_{a_{c,t},x}$  to denote the number of times speaker  $a_{c,t}$  is assigned the topic shift with value  $x \in \{0, 1\}$ ;  $J_{c,s}^x$  to denote the number of topics in segment  $s$  of conversation  $c$  if  $l_{c,t} = x$  and  $N_{c,s,j}^x$  to denote the number of tokens assigned to the segment-specific topic  $j$  when  $l_{c,t} = x$ .<sup>4</sup> Again, the superscript  $^{-c,t}$  is used to denote exclusion of turn  $t$  of conversation  $c$  in the corresponding counts.

Recall that the topic shift is a binary variable. We use 0 to represent the case that the topic distribution is identical to the previous turn. We sample this assignment  $P(l_{c,t} = 0 \mid \mathbf{l}^{-c,t}, \mathbf{w}, \mathbf{k}, \mathbf{a}, *) \propto$

$$\frac{S_{a_{c,t},0}^{-c,t} + \gamma}{S_{a_{c,t},\cdot}^{-c,t} + 2\gamma} \times \frac{\alpha_c^{J_{c,s_t}^0} \prod_{j=1}^{J_{c,s_t}^0} (N_{c,s_t,j}^0 - 1)!}{\prod_{x=1}^{N_{c,s_t,\cdot}^0} (x - 1 + \alpha_c)}. \quad (3)$$

<sup>4</sup>Deterministically knowing the path assignments is the primary efficiency motivation for using the minimal path assumption. The alternative is to explicitly sample the path assignments, which is more complicated (for both notation and computation). This option is spelled in full detail in the supplementary material.

In Equation 3, the first factor is proportional to the probability of assigning a topic shift of value 0 to speaker  $a_{c,t}$  and the second factor is proportional to the joint probability of all topics in segment  $s_t$  of conversation  $c$  when  $l_{c,t} = 0$ .

The other alternative is for the topic shift to be 1, which represents the introduction of a new distribution over topics *inside* an existing segment. We sample this as  $P(l_{c,t} = 1 \mid \mathbf{l}^{-c,t}, \mathbf{w}, \mathbf{k}, \mathbf{a}, *) \propto$

$$\frac{S_{a_{c,t},1}^{-c,t} + \gamma}{S_{a_{c,t},\cdot}^{-c,t} + 2\gamma} \times \left( \frac{\alpha_c^{J_{c,(s_t-1)}^1} \prod_{j=1}^{J_{c,(s_t-1)}^1} (N_{c,(s_t-1),j}^1 - 1)!}{\prod_{x=1}^{N_{c,(s_t-1),\cdot}^1} (x - 1 + \alpha_c)} \frac{\alpha_c^{J_{c,s_t}^1} \prod_{j=1}^{J_{c,s_t}^1} (N_{c,s_t,j}^1 - 1)!}{\prod_{x=1}^{N_{c,s_t,\cdot}^1} (x - 1 + \alpha_c)} \right). \quad (4)$$

As above, the first factor in Equation 4 is proportional to the probability of assigning a topic shift of value 1 to speaker  $a_{c,t}$ ; the second factor in the big bracket is proportional to the joint distribution of the topics in segments  $s_t - 1$  and  $s_t$ . In this case  $l_{c,t} = 1$  means splitting the current segment, which results in two joint probabilities for two segments.

## 4 Datasets

This section introduces the three corpora we use. We preprocess the data to remove stopwords and remove turns containing fewer than five tokens.

**The ICSI Meeting Corpus:** The *ICSI Meeting Corpus* (Janin et al., 2003) is 75 transcribed meetings. For evaluation, we used a standard set of reference segmentations (Galley et al., 2003) of 25 meetings. Segmentations are *binary*, i.e., each point of the document is either a segment boundary or not, and on average each meeting has 8 segment boundaries. After preprocessing, there are 60 unique speakers and the vocabulary contains 3346 non-stopword tokens.

**The 2008 Presidential Election Debates** Our second dataset contains three annotated presidential debates (Boydston et al., 2011) between Barack Obama and John McCain and a vice presidential debate between Joe Biden and Sarah Palin. Each turn is one of two types: *questions* ( $Q$ ) from the moderator or *responses* ( $R$ ) from a candidate. Each clause in a turn is coded with a *Question Topic* ( $T_Q$ ) and a *Response Topic* ( $T_R$ ). Thus, a turn has a list of  $T_Q$ 's and  $T_R$ 's both of length equal to the number of clauses in the turn. Topics are from the Policy Agendas Topics

Speaker	Type	Turn clauses	$T_Q$	$T_R$
Brokaw	$Q$	Sen. Obama, [...] Are you saying [...] that the American economy is going to get much worse before it gets better and they ought to be prepared for that?	1	N/A
Obama	$R$	No, I am confident about the American economy.	1	1
		[...] But most importantly, we're going to have to help ordinary families be able to stay in their homes, make sure that they can pay their bills [...]	1	14
Brokaw	$Q$	Sen. McCain, in all candor, do you think the economy is going to get worse before it gets better?	1	N/A
McCain	$R$	[...] I think if we act effectively, if we stabilize the housing market—which I believe we can,	1	14
		if we go out and buy up these bad loans, so that people can have a new mortgage at the new value of their home	1	14
		I think if we get rid of the cronyism and special interest influence in Washington so we can act more effectively. [...]	1	20

Table 1: Example turns from the annotated 2008 election debates. The topics ( $T_Q$  and  $T_R$ ) are from the Policy Agendas Topics Codebook which contains the following codes of topic: Macroeconomics (1), Housing & Community Development (14), Government Operations (20).

Codebook, a manual inventory of 19 major topics and 225 subtopics.<sup>5</sup> Table 1 shows an example annotation.

To get reference segmentations, we assign each turn a real value from 0 to 1 indicating how much a turn changes the topic. For a question-typed turn, the score is the fraction of clause topics not appearing in the previous turn; for response-typed turns, the score is the fraction of clause topics that do not appear in the corresponding question. This results in a set of *non-binary* reference segmentations. For evaluation metrics that require binary segmentations, we create a binary segmentation by setting a turn as a segment boundary if the computed score is 1. This threshold is chosen to include only true segment boundaries.

**CNN’s Crossfire** *Crossfire* was a weekly U.S. television “talking heads” program engineered to incite heated arguments (hence the name). Each episode features two recurring hosts, two guests, and clips from the week’s news. Our *Crossfire* dataset contains 1134 transcribed episodes aired between 2000 and 2004.<sup>6</sup> There are 2567 unique speakers. Unlike the previous two datasets, *Crossfire* does not have explicit topic segmentations, so we use it to explore speaker-specific characteristics (Section 6).

## 5 Topic Segmentation Experiments

In this section, we examine how well SITS can replicate annotations of when new topics are introduced.

<sup>5</sup> <http://www.policyagendas.org/page/topic-codebook>

<sup>6</sup> <http://www.cs.umd.edu/~vietan/topicshift/crossfire.zip>

We discuss metrics for evaluating an algorithm’s segmentation against a gold annotation, describe our experimental setup, and report those results.

**Evaluation Metrics** To evaluate segmentations, we use  $P_k$  (Beeferman et al., 1999) and WindowDiff (WD) (Pevzner and Hearst, 2002). Both metrics measure the probability that two points in a document will be incorrectly separated by a segment boundary. Both techniques consider all spans of length  $k$  in the document and count whether the two endpoints of the window are (im)properly segmented against the gold segmentation.

However, these metrics have drawbacks. First, they require both hypothesized and reference segmentations to be binary. Many algorithms (e.g., probabilistic approaches) give *non-binary* segmentations where candidate boundaries have real-valued scores (e.g., probability or confidence). Thus, evaluation requires arbitrary thresholding to binarize soft scores. To be fair, thresholds are set so the number of segments are equal to a predefined value (Purver et al., 2006; Galley et al., 2003).

To overcome these limitations, we also use *Earth Mover’s Distance* (EMD) (Rubner et al., 2000), a metric that measures the distance between two distributions. The EMD is the minimal cost to transform one distribution into the other. Each segmentation can be considered a multi-dimensional distribution where each candidate boundary is a dimension. In EMD, a distance function across features allows partial credit for “near miss” segment boundaries. In

addition, because EMD operates on distributions, we can compute the distance between non-binary hypothesized segmentations with binary or real-valued reference segmentations. We use the FastEMD implementation (Pele and Werman, 2009).

**Experimental Methods** We applied the following methods to discover topic segmentations in a document:

- **TextTiling** (Hearst, 1997) is one of the earliest general-purpose topic segmentation algorithms, sliding a fixed-width window to detect major changes in lexical similarity.
- **P-NoSpeaker-S**: parametric version without speaker identity run on each conversation (Purver et al., 2006)
- **P-NoSpeaker-M**: parametric version without speaker identity run on all conversations
- **P-SITS**: the parametric version of SITS with speaker identity run on all conversations
- **NP-HMM**: the HMM-based nonparametric model which a single topic per turn. This model can be considered a Sticky HDP-HMM (Fox et al., 2008) with speaker identity.
- **NP-SITS**: the nonparametric version of SITS with speaker identity run on all conversations.

**Parameter Settings and Implementations** In our experiment, all parameters of TextTiling are the same as in (Hearst, 1997). For statistical models, Gibbs sampling with 10 randomly initialized chains is used. Initial hyperparameter values are sampled from  $U(0, 1)$  to favor sparsity; statistics are collected after 500 burn-in iterations with a lag of 25 iterations over a total of 5000 iterations; and slice sampling (Neal, 2003) optimizes hyperparameters.

**Results and Analysis** Table 2 shows the performance of various models on the topic segmentation problem, using the ICSI corpus and the 2008 debates.

Consistent with previous results, probabilistic models outperform TextTiling. In addition, among the probabilistic models, the models that had access to speaker information consistently segment better than those lacking such information, supporting our assertion that there is benefit to modeling conversation as a *social* process. Furthermore, NP-SITS outperforms NP-HMM in both experiments, suggesting that using a distribution over topics to turns is better than using a single topic. This is consistent with parametric results reported in (Purver et al., 2006).

The contribution of speaker identity seems more valuable in the debate setting. Debates are characterized by strong rewards for setting the agenda; dodging a question or moving the debate toward an oppo-

nent’s weakness can be useful strategies (Boydston et al., 2011). In contrast, meetings (particularly low-stakes ICSI meetings) are characterized by pragmatic rather than strategic topic shifts. Second, agenda-setting roles are clearer in formal debates; a moderator is tasked with setting the agenda and ensuring the conversation does not wander too much.

The nonparametric model does best on the smaller debate dataset. We suspect that an evaluation that directly accessed the topic quality, either via prediction (Teh et al., 2006) or interpretability (Chang et al., 2009) would favor the nonparametric model more.

## 6 Evaluating Topic Shift Tendency

In this section, we focus on the ability of SITS to capture speaker-level attributes. Recall that SITS associates with each speaker a topic shift tendency  $\pi$  that represents the probability of asserting a new topic in the conversation. While topic segmentation is a well studied problem, there are no established quantitative measurements of an individual’s ability to control a conversation. To evaluate whether the tendency is capturing meaningful characteristics of speakers, we compare our inferred tendencies against insights from political science.

**2008 Elections** To obtain a posterior estimate of  $\pi$  (Figure 3) we create 10 chains with hyperparameters sampled from the uniform distribution  $U(0, 1)$  and averaged  $\pi$  over 10 chains (as described in Section 5). In these debates, Ifill is the moderator of the debate between Biden and Palin; Brokaw, Lehrer and Schieffer are the three moderators of three debates between Obama and McCain. Here “Question” denotes questions from audiences in “town hall” debate. The role of this “speaker” can be considered equivalent to the debate moderator.

The topic shift tendencies of moderators are much higher than for candidates. In the three debates between Obama and McCain, the moderators—Brokaw, Lehrer and Schieffer—have significantly higher scores than both candidates. This is a useful reality check, since in a debate the moderators are the ones asking questions and literally controlling the topical focus. Interestingly, in the vice-presidential debate, the score of moderator Ifill is only slightly higher than those of Palin and Biden; this is consistent with media commentary characterizing her as a

	Model	EMD	$P_k$			WindowDiff		
			$k = 5$	10	15	$k = 5$	10	15
ICSI Dataset	TextTiling	2.507	.289	.388	.451	.318	.477	.561
	P-NoSpeaker-S	1.949	.222	.283	.342	.269	.393	.485
	P-NoSpeaker-M	1.935	<b>.207</b>	.279	.335	<b>.253</b>	.371	.468
	P-SITS	<b>1.807</b>	.211	<b>.251</b>	.289	.256	<b>.363</b>	<b>.434</b>
	NP-HMM	2.189	.232	.257	.263	.267	.377	.444
	NP-SITS	2.126	.228	.253	<b>.259</b>	.262	.372	.440
Debates Dataset	TextTiling	2.821	.433	.548	.633	.534	.674	.760
	P-NoSpeaker-S	2.822	.426	.543	.653	.482	.650	.756
	P-NoSpeaker-M	2.712	.411	.522	.589	.479	.644	.745
	P-SITS	2.269	.380	.405	.402	.482	.625	.719
	NP-HMM	2.132	.362	.348	.323	.486	.629	.723
	NP-SITS	<b>1.813</b>	<b>.332</b>	<b>.269</b>	<b>.231</b>	<b>.470</b>	<b>.600</b>	<b>.692</b>

Table 2: Results on the topic segmentation task. Lower is better. The parameter  $k$  is the window size of the metrics  $P_k$  and WindowDiff chosen to replicate previous results.

weak moderator.<sup>7</sup> Similarly, the “Question” speaker had a relatively high variance, consistent with an amalgamation of many distinct speakers.

These topic shift tendencies suggest that all candidates manage to succeed at some points in setting and controlling the debate topics. Our model gives Obama a slightly higher score than McCain, consistent with social science claims (Boydston et al., 2011) that Obama had the lead in setting the agenda over McCain. Table 4 shows of SITS-detected topic shifts.

**Crossfire** Crossfire, unlike the debates, has many speakers. This allows us to examine more closely what we can learn about speakers’ topic shift tendency. We verified that SITS can segment topics, and assuming that changing the topic is useful for a speaker, how can we characterize who does so effectively? We examine the relationship between topic shift tendency, social roles, and political ideology.

To focus on frequent speakers, we filter out speakers with fewer than 30 turns. Most speakers have relatively small  $\pi$ , with the mode around 0.3. There are, however, speakers with very high topic shift tendencies. Table 5 shows the speakers having the highest values according to SITS.

We find that there are three general patterns for who influences the course of a conversation in *Crossfire*. First, there are structural “speakers” the show uses to frame and propose new topics. These are

<sup>7</sup> <http://harpers.org/archive/2008/10/hbc-90003659>

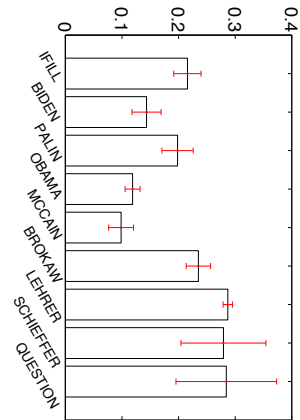


Table 3: Topic shift tendency  $\pi$  of speakers in the 2008 Presidential Election Debates (larger means greater tendency)

audience questions, news clips (e.g. many of Gore’s and Bush’s turns from 2000), and voice overs. That SITS is able to recover these is reassuring. Second, the stable of regular hosts receives high topic shift tendencies, which is reasonable given their experience with the format and ostensible moderation roles (in practice they also stoke lively discussion).

The remaining class is more interesting. The remaining non-hosts with high topic shift tendency are relative moderates on the political spectrum:

- John Kasich, one of few Republicans to support the assault weapons ban and now governor of Ohio, a swing state
- Christine Todd Whitman, former Republican governor of New Jersey, a very Democratic state
- John McCain, who before 2008 was known as a “maverick” for working with Democrats (e.g. Russ Feingold)

This suggests that, despite *Crossfire*’s tendency to create highly partisan debates, those who are able to work across the political spectrum may best be able to influence the topic under discussion in highly polarized contexts. Table 4 shows detected topic shifts from these speakers; two of these examples (McCain and Whitman) show disagreement of Republicans with President Bush. In the other, Kasich is defending a Republican plan (school vouchers) popular with traditional Democratic constituencies.

## 7 Related and Future Work

In the realm of statistical models, a number of techniques incorporate social connections and identity to explain content in social networks (Chang and Blei,



	Previous turn	Turn detected as shifting topic
Debates Dataset	PALIN: Your question to him was whether he supported <i>gay marriage</i> and my answer is the same as his and it is that I do not.	IFILL: Wonderful. You agree. On that note, let’s move to <i>foreign policy</i> . You both have sons who are in Iraq or on their way to Iraq. You, Governor Palin, have said that you would like to see a real clear plan for an exit strategy. [...]
	MCCAIN: I think that Joe Biden is qualified in many respects. ...	SCHIEFFER: [...] Let’s talk about <i>energy</i> and <i>climate control</i> . Every president since Nixon has said what both of you [...]
	IFILL: So, Governor, as vice president, there’s nothing that you have promised [...] that you wouldn’t take off the table because of this <i>financial crisis</i> we’re in?	BIDEN: Again, let me—let’s talk about those tax breaks. [Obama] voted for an energy bill because, for the first time, it had real support for alternative energy. [...] on eliminating the tax breaks for the oil companies, Barack Obama voted to eliminate them. [...]
Crossfire Dataset	PRESS: But what do you say, governor, to Governor Bush and [...] your party who would let politicians and not medical scientists decide what <i>drugs</i> are distributed [...]	WHITMAN: Well I disagree with them on this particular issues [...] that’s important to me that George Bush stands for education of our children [...] I care about tax policy, I care about the environment. I care about all the issues where he has a proven record in Texas [...]
	WEXLER: [...] They need a Medicare prescription drug plan [...] Talk about schools, [...] Al Gore has got a real plan. George Bush offers us vouchers. Talk about the environment. [...] Al Gore is right on in terms of the majority of Americans, but George Bush [...]	KASICH: [...] I want to talk about choice. [...] George Bush believes that, if schools fail, parents ought to have a right to get their kids out of those schools and give them a chance and an opportunity for success. Gore says “no way” [...] Social Security. George Bush says [...] direct it the way federal employees do [...] Al Gore says “No way” [...] That’s real choice. That’s real bottom-up, not a bureaucratic approach, the way we run this country.
	PRESS: Senator, Senator Breaux mentioned that it’s President Bush’s aim to start on education [...] [McCain] [...] said he was going to do introduce the legislation the first day of the first week of the new administration. [...]	MCCAIN: After one of closest elections in our nation’s history, there is one thing the American people are unanimous about They want their government back. We can do that by ridding politics of large, unregulated contributions that give special interests a seat at the table while average Americans are stuck in the back of the room.

Table 4: Example of turns designated as a topic shift by SITS. Turns were chosen with speakers to give examples of those with high topic shift tendency  $\pi$ .

Rank	Speaker	$\pi$	Rank	Speaker	$\pi$
1	Announcer	.884	10	Kasich	.570
2	Male	.876	11	Carville <sup>†</sup>	.550
3	Question	.755	12	Carlson <sup>†</sup>	.550
4	G. W. Bush <sup>‡</sup>	.751	13	Begala <sup>†</sup>	.545
5	Press <sup>†</sup>	.651	14	Whitman	.533
6	Female	.650	15	McAuliffe	.529
7	Gore <sup>‡</sup>	.650	16	Matalin <sup>†</sup>	.527
8	Narrator	.642	17	McCain	.524
9	Novak <sup>†</sup>	.587	18	Fleischer	.522

Table 5: Top speakers by topic shift tendencies. We mark hosts (†) and “speakers” who often (but not always) appeared in clips (‡). Apart from those groups, speakers with the highest tendency were political moderates.

2009) and scientific corpora (Rosen-Zvi et al., 2004). However, these models ignore the temporal evolution of content, treating documents as static.

Models that do investigate the evolution of topics over time typically ignore the identify of the speaker. For example: models having sticky topics over n-grams (Johnson, 2010), sticky HDP-HMM (Fox et al., 2008); models that are an amalgam of sequential models and topic models (Griffiths et al., 2005; Wal-

lach, 2006; Gruber et al., 2007; Ahmed and Xing, 2008; Boyd-Graber and Blei, 2008; Du et al., 2010); or explicit models of time or other relevant features as a distinct latent variable (Wang and McCallum, 2006; Eisenstein et al., 2010).

In contrast, SITS jointly models topic and individuals’ tendency to control a conversation. Not only does SITS outperform other models using standard computational linguistics baselines, but it also proposes intriguing hypotheses for social scientists.

Associating each speaker with a scalar that models their tendency to change the topic does improve performance on standard tasks, but it’s inadequate to fully describe an individual. Modeling individuals’ perspective (Paul and Girju, 2010), “side” (Thomas et al., 2006), or personal preferences for topics (Grimmer, 2009) would enrich the model and better illuminate the interaction of influence and topic.

Statistical analysis of political discourse can help discover patterns that political scientists, who often work via a “close reading,” might otherwise miss. We plan to work with social scientists to validate our implicit hypothesis that our topic shift tendency correlates well with intuitive measures of “influence.”

## Acknowledgements

This research was funded in part by the Army Research Laboratory through ARL Cooperative Agreement W911NF-09-2-0072 and by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the Army Research Laboratory. Jordan Boyd-Graber and Philip Resnik are also supported by US National Science Foundation Grant NSF grant #1018625. Any opinions, findings, conclusions, or recommendations expressed are the authors' and do not necessarily reflect those of the sponsors.

## References

- [Abbott et al., 2011] Abbott, R., Walker, M., Anand, P., Fox Tree, J. E., Bowmani, R., and King, J. (2011). How can you say such things?!?: Recognizing disagreement in informal political argument. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, pages 2–11.
- [Ahmed and Xing, 2008] Ahmed, A. and Xing, E. P. (2008). Dynamic non-parametric mixture models and the recurrent Chinese restaurant process: with applications to evolutionary clustering. In *SDM*, pages 219–230.
- [Beeferman et al., 1999] Beeferman, D., Berger, A., and Lafferty, J. (1999). Statistical models for text segmentation. *Mach. Learn.*, 34:177–210.
- [Blei and Lafferty, 2009] Blei, D. M. and Lafferty, J. (2009). *Text Mining: Theory and Applications*, chapter Topic Models. Taylor and Francis, London.
- [Boyd-Graber and Blei, 2008] Boyd-Graber, J. and Blei, D. M. (2008). Syntactic topic models. In *Proceedings of Advances in Neural Information Processing Systems*.
- [Boydston et al., 2011] Boydston, A. E., Phillips, C., and Glazier, R. A. (2011). It's the economy again, stupid: Agenda control in the 2008 presidential debates. *Forthcoming*.
- [Chang and Blei, 2009] Chang, J. and Blei, D. M. (2009). Relational topic models for document networks. In *Proceedings of Artificial Intelligence and Statistics*.
- [Chang et al., 2009] Chang, J., Boyd-Graber, J., Wang, C., Gerrish, S., and Blei, D. M. (2009). Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems*.
- [Du et al., 2010] Du, L., Buntine, W., and Jin, H. (2010). Sequential latent dirichlet allocation: Discover underlying topic structures within a document. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 148–157.
- [Ehlen et al., 2007] Ehlen, P., Purver, M., and Niekrazz, J. (2007). A meeting browser that learns. In *In: Proceedings of the AAAI Spring Symposium on Interaction Challenges for Intelligent Assistants*.
- [Eisenstein and Barzilay, 2008] Eisenstein, J. and Barzilay, R. (2008). Bayesian unsupervised topic segmentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Proceedings of Empirical Methods in Natural Language Processing.
- [Eisenstein et al., 2010] Eisenstein, J., O'Connor, B., Smith, N. A., and Xing, E. P. (2010). A latent variable model for geographic lexical variation. In *EMNLP'10*, pages 1277–1287.
- [Ferguson, 1973] Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230.
- [Fox et al., 2008] Fox, E. B., Sudderth, E. B., Jordan, M. I., and Willsky, A. S. (2008). An hdp-hmm for systems with state persistence. In *Proceedings of International Conference of Machine Learning*.
- [Galley et al., 2003] Galley, M., McKeown, K., Fosler-Lussier, E., and Jing, H. (2003). Discourse segmentation of multi-party conversation. In *Proceedings of the Association for Computational Linguistics*.
- [Georgescul et al., 2006] Georgescul, M., Clark, A., and Armstrong, S. (2006). Word distributions for thematic segmentation in a support vector machine approach. In *Conference on Computational Natural Language Learning*.
- [Gerrish and Blei, 2010] Gerrish, S. and Blei, D. M. (2010). A language-based approach to measuring scholarly impact. In *Proceedings of International Conference of Machine Learning*.
- [Griffiths et al., 2005] Griffiths, T. L., Steyvers, M., Blei, D. M., and Tenenbaum, J. B. (2005). Integrating topics and syntax. In *Proceedings of Advances in Neural Information Processing Systems*.
- [Grimmer, 2009] Grimmer, J. (2009). A Bayesian Hierarchical Topic Model for Political Texts: Measuring Expressed Agendas in Senate Press Releases. *Political Analysis*, 18:1–35.
- [Gruber et al., 2007] Gruber, A., Rosen-Zvi, M., and Weiss, Y. (2007). Hidden topic Markov models. In *Artificial Intelligence and Statistics*.
- [Hawes et al., 2009] Hawes, T., Lin, J., and Resnik, P. (2009). Elements of a computational model for multi-party discourse: The turn-taking behavior of Supreme Court justices. *Journal of the American Society for Information Science and Technology*, 60(8):1607–1615.
- [Hearst, 1997] Hearst, M. A. (1997). TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.

- [Hsueh et al., 2006] Hsueh, P.-y., Moore, J. D., and Renals, S. (2006). Automatic segmentation of multiparty dialogue. In *Proceedings of the European Chapter of the Association for Computational Linguistics*.
- [Ireland et al., 2011] Ireland, M. E., Slatcher, R. B., Eastwick, P. W., Scissors, L. E., Finkel, E. J., and Pennebaker, J. W. (2011). Language style matching predicts relationship initiation and stability. *Psychological Science*, 22(1):39–44.
- [Janin et al., 2003] Janin, A., Baron, D., Edwards, J., Ellis, D., Gelbart, D., Morgan, N., Peskin, B., Pfau, T., Shriberg, E., Stolcke, A., and Wooters, C. (2003). The ICSI meeting corpus. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*.
- [Johnson, 2010] Johnson, M. (2010). PCFGs, topic models, adaptor grammars and learning topical collocations and the structure of proper names. In *Proceedings of the Association for Computational Linguistics*.
- [Morris and Hirst, 1991] Morris, J. and Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17:21–48.
- [Müller and Quintana, 2004] Müller, P. and Quintana, F. A. (2004). Nonparametric Bayesian data analysis. *Statistical Science*, 19(1):95–110.
- [Murray et al., 2005] Murray, G., Renals, S., and Carletta, J. (2005). Extractive summarization of meeting recordings. In *European Conference on Speech Communication and Technology*.
- [Neal, 2000] Neal, R. M. (2000). Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265.
- [Neal, 2003] Neal, R. M. (2003). Slice sampling. *Annals of Statistics*, 31:705–767.
- [Olney and Cai, 2005] Olney, A. and Cai, Z. (2005). An orthonormal basis for topic segmentation in tutorial dialogue. In *Proceedings of the Human Language Technology Conference*.
- [Paul and Girju, 2010] Paul, M. and Girju, R. (2010). A two-dimensional topic-aspect model for discovering multi-faceted topics. In *Association for the Advancement of Artificial Intelligence*.
- [Pele and Werman, 2009] Pele, O. and Werman, M. (2009). Fast and robust earth mover’s distances. In *International Conference on Computer Vision*.
- [Pevzner and Hearst, 2002] Pevzner, L. and Hearst, M. A. (2002). A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28.
- [Purver, 2011] Purver, M. (2011). Topic segmentation. In Tur, G. and de Mori, R., editors, *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*, pages 291–317. Wiley.
- [Purver et al., 2006] Purver, M., Körding, K., Griffiths, T. L., and Tenenbaum, J. (2006). Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of the Association for Computational Linguistics*.
- [Resnik and Hardisty, 2010] Resnik, P. and Hardisty, E. (2010). Gibbs sampling for the uninitiated. Technical Report UMIACS-TR-2010-04, University of Maryland. <http://www.lib.umd.edu/drum/handle/1903/10058>.
- [Reynar, 1998] Reynar, J. C. (1998). *Topic Segmentation: Algorithms and Applications*. PhD thesis, University of Pennsylvania.
- [Rosen-Zvi et al., 2004] Rosen-Zvi, M., Griffiths, T. L., Steyvers, M., and Smyth, P. (2004). The author-topic model for authors and documents. In *Proceedings of Uncertainty in Artificial Intelligence*.
- [Rubner et al., 2000] Rubner, Y., Tomasi, C., and Guibas, L. J. (2000). The earth mover’s distance as a metric for image retrieval. *International Journal of Computer Vision*, 40:99–121.
- [Teh et al., 2006] Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.
- [Thomas et al., 2006] Thomas, M., Pang, B., and Lee, L. (2006). Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. In *Proceedings of Empirical Methods in Natural Language Processing*.
- [Tur et al., 2010] Tur, G., Stolcke, A., Voss, L., Peters, S., Hakkani-Tür, D., Dowding, J., Favre, B., Fernández, R., Frampton, M., Frandsen, M., Frederickson, C., Graciana, M., Kintzing, D., Leveque, K., Mason, S., Niekrasz, J., Purver, M., Riedhammer, K., Shriberg, E., Tien, J., Vergyri, D., and Yang, F. (2010). The CALO meeting assistant system. *Trans. Audio, Speech and Lang. Proc.*, 18:1601–1611.
- [Wallach, 2006] Wallach, H. M. (2006). Topic modeling: Beyond bag-of-words. In *Proceedings of International Conference of Machine Learning*.
- [Wallach, 2008] Wallach, H. M. (2008). *Structured Topic Models for Language*. PhD thesis, University of Cambridge.
- [Wang et al., 2008] Wang, C., Blei, D. M., and Heckerman, D. (2008). Continuous time dynamic topic models. In *Proceedings of Uncertainty in Artificial Intelligence*.
- [Wang and McCallum, 2006] Wang, X. and McCallum, A. (2006). Topics over time: a non-Markov continuous-time model of topical trends. In *Knowledge Discovery and Data Mining*, Knowledge Discovery and Data Mining.

# Extracting Narrative Timelines as Temporal Dependency Structures

**Oleksandr Kolomiyets**  
KU Leuven  
Celestijnenlaan 200A  
B-3001 Heverlee, Belgium  
Oleksandr.Kolomiyets@  
cs.kuleuven.be

**Steven Bethard**  
University of Colorado  
Campus Box 594  
Boulder, CO 80309, USA  
Steven.Bethard@  
colorado.edu

**Marie-Francine Moens**  
KU Leuven  
Celestijnenlaan 200A  
B-3001 Heverlee, Belgium  
Sien.Moens@  
cs.kuleuven.be

## Abstract

We propose a new approach to characterizing the timeline of a text: temporal dependency structures, where all the events of a narrative are linked via partial ordering relations like BEFORE, AFTER, OVERLAP and IDENTITY. We annotate a corpus of children’s stories with temporal dependency trees, achieving agreement (Krippendorff’s Alpha) of 0.856 on the event words, 0.822 on the links between events, and of 0.700 on the ordering relation labels. We compare two parsing models for temporal dependency structures, and show that a deterministic non-projective dependency parser outperforms a graph-based maximum spanning tree parser, achieving labeled attachment accuracy of 0.647 and labeled tree edit distance of 0.596. Our analysis of the dependency parser errors gives some insights into future research directions.

## 1 Introduction

There has been much recent interest in identifying events, times and their relations along the timeline, from event and time ordering problems in the TempEval shared tasks (Verhagen et al., 2007; Verhagen et al., 2010), to identifying time arguments of event structures in the Automated Content Extraction program (Linguistic Data Consortium, 2005; Gupta and Ji, 2009), to timestamping event intervals in the Knowledge Base Population shared task (Artiles et al., 2011; Amigó et al., 2011).

However, to date, this research has produced fragmented document timelines, because only specific types of temporal relations in specific contexts have

been targeted. For example, the TempEval tasks only looked at relations between events in the same or adjacent sentences (Verhagen et al., 2007; Verhagen et al., 2010), and the Automated Content Extraction program only looked at time arguments for specific types of events, like *being born* or *transferring money*.

In this article, we propose an approach to temporal information extraction that identifies a single connected timeline for a text. The temporal language in a text often fails to specify a total ordering over all the events, so we annotate the timelines as temporal dependency structures, where each event is a node in the dependency tree, and each edge between nodes represents a temporal ordering relation such as BEFORE, AFTER, OVERLAP or IDENTITY. We construct an evaluation corpus by annotating such temporal dependency trees over a set of children’s stories. We then demonstrate how to train a timeline extraction system based on dependency parsing techniques instead of the pair-wise classification approaches typical of prior work.

The main contributions of this article are:

- We propose a new approach to characterizing temporal structure via dependency trees.
- We produce an annotated corpus of temporal dependency trees in children’s stories.
- We design a non-projective dependency parser for inferring timelines from text.

The following sections first review some relevant prior work, then describe the corpus annotation and the dependency parsing algorithm, and finally present our evaluation results.

## 2 Related Work

Much prior work on the annotation of temporal information has constructed corpora with incomplete timelines. The TimeBank (Pustejovsky et al., 2003b; Pustejovsky et al., 2003a) provided a corpus annotated for all events and times, but temporal relations were only annotated when the relation was judged to be salient by the annotator. In the TempEval competitions (Verhagen et al., 2007; Verhagen et al., 2010), annotated texts were provided for a few different event and time configurations, for example, an event and a time in the same sentence, or two main-clause events from adjacent sentences. Bethard et al. (2007) proposed to annotate temporal relations one syntactic construction at a time, producing an initial corpus of only verbal events linked to events in subordinated clauses. One notable exception to this pattern of incomplete timelines is the work of Bramsen et al. (2006) where temporal structures were annotated as directed acyclic graphs. However they worked on a much coarser granularity, annotating not the ordering between individual events, but between multi-sentence segments of text.

In part because of the structure of the available training corpora, most existing temporal information extraction models formulate temporal linking as a pair-wise classification task, where each pair of events and/or times is examined and classified as having a temporal relation or not. Early work on the TimeBank took this approach (Boguraev and Ando, 2005), classifying relations between all events and times within 64 tokens of each other. Most of the top-performing systems in the TempEval competitions also took this pair-wise classification approach for both event-time and event-event temporal relations (Bethard and Martin, 2007; Cheng et al., 2007; UzZaman and Allen, 2010; Llorens et al., 2010). Systems have also tried to take advantage of more global information to ensure that the pair-wise classifications satisfy temporal logic transitivity constraints, using frameworks such as integer linear programming and Markov logic networks (Bramsen et al., 2006; Chambers and Jurafsky, 2008; Yoshikawa et al., 2009; UzZaman and Allen, 2010). Yet the basic approach is still centered around pair-wise classifications, not the complete temporal structure of a document.

Our work builds upon this prior research, both

improving the annotation approach to generate the fully connected timeline of a story, and improving the models for timeline extraction using dependency parsing techniques. We use the annotation scheme introduced in more detail in Bethard et al. (2012), which proposes to annotate temporal relations as dependency links between head events and dependent events. This annotation scheme addresses the issues of incoherent and incomplete annotations by guaranteeing that all events in a plot are connected along a single timeline. These connected timelines allow us to design new models for timeline extraction in which we jointly infer the temporal structure of the text and the labeled temporal relations. We employ methods from syntactic dependency parsing, adapting them to our task by including features typical of temporal relation labeling models.

## 3 Corpus Annotation

The corpus of stories for children was drawn from the fables collection of (McIntyre and Lapata, 2009)<sup>1</sup> and annotated as described in (Bethard et al., 2012). In this section we illustrate the main annotation principles for coherent temporal annotation. As an example story, consider:

Two Travellers were on the road together, when a Bear suddenly appeared on the scene. Before he observed them, one made for a tree at the side of the road, and climbed up into the branches and hid there. The other was not so nimble as his companion; and, as he could not escape, he threw himself on the ground and pretended to be dead. . . [37.txt]

Figure 1 shows the temporal dependency structure that we expect our annotators to identify in this story.

The annotators were provided with guidelines both for which kinds of words should be identified as events, and for which kinds of events should be linked by temporal relations. For identifying event words, the standard TimeML guidelines for annotating events (Pustejovsky et al., 2003a) were augmented with two additional guidelines:

<sup>1</sup>Data available at <http://homepages.inf.ed.ac.uk/s0233364/McIntyreLapata09/>

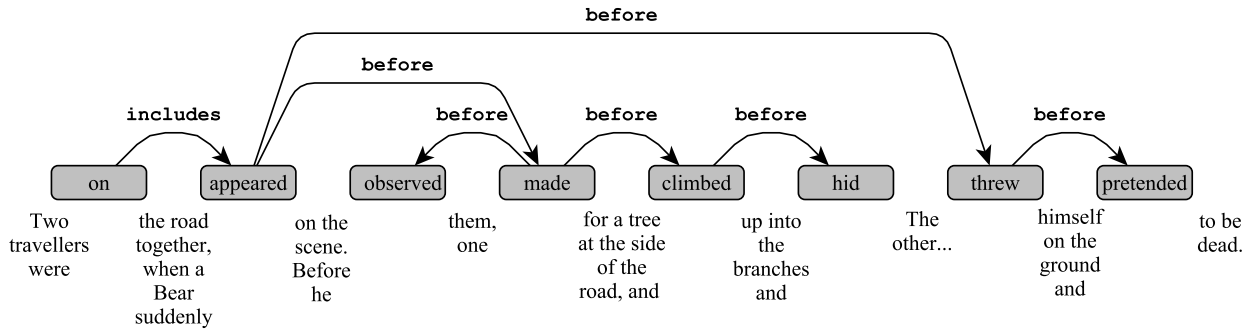


Figure 1: Event timeline for the story of the Travellers and the Bear. Nodes are events and edges are temporal relations. Edges denote temporal relations signaled by linguistic cues in the text. Temporal relations that can be inferred via transitivity are not shown.

- Skip negated, modal or hypothetical events (e.g. *could not escape*, *dead in pretended to be dead*).
- For phrasal events, select the single word that best paraphrases the meaning (e.g. in *used to snap* the event should be *snap*, in *kept perfectly still* the event should be *still*).

For identifying the temporal dependencies (i.e. the ordering relations between event words), the annotators were instructed to link each event in the story to a single nearby event, similar to what has been observed in reading comprehension studies (Johnson-Laird, 1980; Brewer and Lichtenstein, 1982). When there were several reasonable nearby events to choose from, the annotators were instructed to choose the temporal relation that was easiest to infer from the text (e.g. preferring relations with explicit cue words like *before*). A set of six temporal relations was used: BEFORE, AFTER, INCLUDES, IS-INCLUDED, IDENTITY or OVERLAP.

Two annotators annotated temporal dependency structures in the first 100 fables of the McIntyre-Lapata collection and measured inter-annotator agreement by Krippendorff’s Alpha for nominal data (Krippendorff, 2004; Hayes and Krippendorff, 2007). For the resulting annotated corpus annotators achieved Alpha of 0.856 on the event words, 0.822 on the links between events, and of 0.700 on the ordering relation labels. Thus, we concluded that the temporal dependency annotation paradigm was reliable, and the resulting corpus of 100 fables<sup>2</sup> could be used to

<sup>2</sup>Available from <http://www.bethard.info/data/fables-100-temporal-dependency.xml>

train a temporal dependency parsing model.

## 4 Parsing Models

We consider two different approaches to learning a temporal dependency parser: a shift-reduce model (Nivre, 2008) and a graph-based model (McDonald et al., 2005). Both models take as input a sequence of event words and produce as output a tree structure where the events are linked via temporal relations. Formally, a parsing model is a function ( $W \rightarrow \Pi$ ) where  $W = w_1w_2 \dots w_n$  is a sequence of event words, and  $\pi \in \Pi$  is a dependency tree  $\pi = (V, E)$  where:

- $V = W \cup \{Root\}$ , that is, the vertex set of the graph is the set of words in  $W$  plus an artificial root node.
- $E = \{(w_h, r, w_d) : w_h \in V, w_d \in V, r \in R = \{\text{BEFORE, AFTER, INCLUDES, IS\_INCLUDED, IDENTITY, OVERLAP}\}\}$ , that is, in the edge set of the graph, each edge is a link between a dependent word and its head word, labeled with a temporal relation.
- $(w_h, r, w_d) \in E \implies w_d \neq Root$ , that is, the artificial root node has no head.
- $(w_h, r, w_d) \in E \implies ((w'_h, r', w_d) \in E \implies w_h = w'_h \wedge r = r')$ , that is, for every node there is at most one head and one relation label.
- $E$  contains no (non-empty) subset of arcs  $(w_h, r_i, w_i), (w_i, r_j, w_j), \dots, (w_k, r_l, w_h)$ , that is, there are no cycles in the graph.

SHIFT	Move all of $L_2$ and the head of $Q$ onto $L_1$ $([a_1 \dots a_i], [b_1 \dots b_j], [w_k w_{k+1} \dots], E) \rightarrow ([a_1 \dots a_i b_1 \dots b_j w_k], [], [w_{k+1} \dots], E)$
NO-ARC	Move the head of $L_1$ to the head of $L_2$ $([a_1 \dots a_i a_{i+1}], [b_1 \dots b_j], Q, E) \rightarrow ([a_1 \dots a_i], [a_{i+1} b_1 \dots b_j], Q, E)$
LEFT-ARC	Create a relation where the head of $L_1$ depends on the head of $Q$ Not applicable if $a_{i+1}$ is the root or already has a head, or if there is a path connecting $w_k$ and $a_{i+1}$ $([a_1 \dots a_i a_{i+1}], [b_1 \dots b_j], [w_k \dots], E) \rightarrow ([a_1 \dots a_i], [a_{i+1} b_1 \dots b_j], [w_k \dots], E \cup (w_k, r, a_{i+1}))$
RIGHT-ARC	Create a relation where the head of $Q$ depends on the head of $L_1$ Not applicable if $w_k$ is the root or already has a head, or if there is a path connecting $w_k$ and $a_{i+1}$ $([a_1 \dots a_i a_{i+1}], [b_1 \dots b_j], [w_k \dots], E) \rightarrow ([a_1 \dots a_i], [a_{i+1} b_1 \dots b_j], [w_k \dots], E \cup (a_{i+1}, r, w_k))$

Table 1: Transition system for Covington-style shift-reduce dependency parsers.

#### 4.1 Shift-Reduce Parsing Model

Shift-reduce dependency parsers start with an input queue of unlinked words, and link them into a tree by repeatedly choosing and performing actions like shifting a node to a stack, or popping two nodes from the stack and linking them. Shift-reduce parsers are typically defined in terms of configurations and a transition system, where the configurations describe the current internal state of the parser, and the transition system describes how to get from one state to another. Formally, a deterministic shift-reduce dependency parser is defined as  $(C, T, C_F, \text{INIT}, \text{TREE})$  where:

- $C$  is the set of possible parser configurations  $c_i$
- $T \subseteq (C \rightarrow C)$  is the set of transitions  $t_i$  from one configuration  $c_j$  to another  $c_{j+1}$  allowed by the parser
- $\text{INIT} \in (W \rightarrow C)$  is a function from the input words to an initial parser configuration
- $C_F \subseteq C$  are the set of final parser configurations  $c_F$  where the parser is allowed to terminate
- $\text{TREE} \in (C_F \rightarrow \Pi)$  is a function that extracts a dependency tree  $\pi$  from a final parser state  $c_F$

Given this formalism and an oracle  $o \in (C \rightarrow T)$ , which can choose a transition given the current configuration of the parser, dependency parsing can be accomplished by Algorithm 1. For temporal dependency parsing, we adopt the Covington set of transitions (Covington, 2001) as it allows for parsing the non-projective trees, which may also contain ‘‘crossing’’ edges, that occasionally occur in our annotated corpus. Our parser is therefore defined as:

---

#### Algorithm 1 Deterministic parsing with an oracle.

---

```

 $c \leftarrow \text{INIT}(W)$ 
while  $c \notin C_F$  do
   $t \leftarrow o(c)$ 
   $c \leftarrow t(c)$ 
end while
return  $\text{TREE}(c)$ 

```

---

- $c = (L_1, L_2, Q, E)$  is a parser configuration, where  $L_1$  and  $L_2$  are lists for temporary storage,  $Q$  is the queue of input words, and  $E$  is the set of identified edges of the dependency tree.
- $T = \{\text{SHIFT}, \text{NO-ARC}, \text{LEFT-ARC}, \text{RIGHT-ARC}\}$  is the set of transitions described in Table 1.
- $\text{INIT}(W) = ([\text{Root}], [], [w_1, w_2, \dots, w_n], \emptyset)$  puts all input words on the queue and the artificial root on  $L_1$ .
- $C_F = \{(L_1, L_2, Q, E) \in C : L_1 = \{W \cup \{\text{Root}\}\}, L_2 = Q = \emptyset\}$  accepts final states where the input words have been moved off of the queue and lists and into the edges in  $E$ .
- $\text{TREE}((L_1, L_2, Q, E)) = (W \cup \{\text{Root}\}, E)$  extracts the final dependency tree.

The oracle  $o$  is typically defined as a machine learning classifier, which characterizes a parser configuration  $c$  in terms of a set of features. For temporal dependency parsing, we learn a Support Vector Machine classifier (Yamada and Matsumoto, 2003) using the features described in Section 5.

#### 4.2 Graph-Based Parsing Model

One shortcoming of the shift-reduce dependency parsing approach is that each transition decision

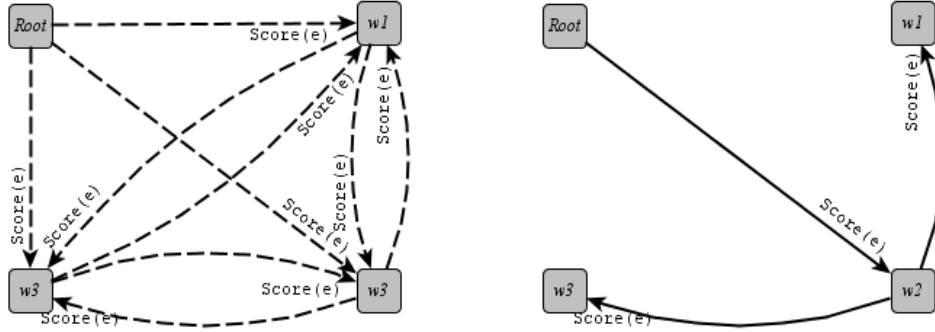


Figure 2: A setting for the graph-based parsing model: an initial dense graph  $G$  (left) with edge scores  $\text{SCORE}(e)$ . The resulting dependency tree as a spanning tree with the highest score over the edges (right).

made by the model is final, and cannot be revisited to search for more globally optimal trees. Graph-based models are an alternative dependency parsing model, which assembles a graph with weighted edges between all pairs of words, and selects the tree-shaped subset of this graph that gives the highest total score (Fig. 2). Formally, a graph-based parser follows Algorithm 2, where:

- $W' = W \cup \{Root\}$
- $\text{SCORE} \in ((W' \times R \times W) \rightarrow \mathfrak{R})$  is a function for scoring edges
- $\text{SPANNINGTREE}$  is a function for selecting a subset of edges that is a tree that spans over all the nodes of the graph.

---

**Algorithm 2** Graph-based dependency parsing

---

$E \leftarrow \{(e, \text{SCORE}(e)) : e \in (W' \times R \times W)\}$   
 $G \leftarrow (W', E)$   
**return**  $\text{SPANNINGTREE}(G)$

---

The  $\text{SPANNINGTREE}$  function is usually defined using one of the efficient search techniques for finding a maximum spanning tree. For temporal dependency parsing, we use the Chu-Liu-Edmonds algorithm (Chu and Liu, 1965; Edmonds, 1967) which solves this problem by iteratively selecting the edge with the highest weight and removing edges that would create cycles. The result is the globally optimal maximum spanning tree for the graph (Georgiadis, 2003).

The  $\text{SCORE}$  function is typically defined as a machine learning model that scores an edge based on a set of features. For temporal dependency parsing, we learn a model to predict edge scores via the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003; Crammer et al., 2006) using the set of features defined in Section 5.

## 5 Feature Design

The proposed parsing algorithms both rely on machine learning methods. The shift-reduce parser (SRP) trains a machine learning classifier as the oracle  $o \in (C \rightarrow T)$  to predict a transition  $t$  from a parser configuration  $c = (L_1, L_2, Q, E)$ , using node features such as the heads of  $L_1$ ,  $L_2$  and  $Q$ , and edge features from the already predicted temporal relations in  $E$ . The graph-based maximum spanning tree (MST) parser trains a machine learning model to predict  $\text{SCORE}(e)$  for an edge  $e = (w_i, r_j, w_k)$ , using features of the nodes  $w_i$  and  $w_k$ . The full set of features proposed for both parsing models, derived from the state-of-the-art systems for temporal relation labeling, is presented in Table 2. Note that both models share features that look at the nodes, while only the shift-reduce parser has features for previously classified edges.

## 6 Evaluations

Evaluations were performed using 10-fold cross-validation on the fables annotated in Section 3. The corpus contains 100 fables, a total of 14,279 tokens and a total of 1136 annotated temporal relations. As



Feature	SRP	MST
Word	√*	√*
Lemma	√*	√*
Part of speech (POS) tag	√*	√*
Suffixes	√*	√*
Syntactically governing verb	√*	√*
Governing verb lemma	√*	√*
Governing verb POS tag	√*	√*
Governing verb POS suffixes	√*	√*
Prepositional phrase occurrence	√*	√*
Dominated by auxiliary verb?	√*	√*
Dominated by modal verb?	√*	√*
Temporal signal word is nearby?	√*	√*
Head word lemma	√*	√*
Temporal relation labels of $a_i$ and its leftmost and rightmost dependents	√	
Temporal relation labels of $a_{i-1}$ 's leftmost and rightmost dependents	√	
Temporal relation labels of $b_1$ and its leftmost and rightmost dependents	√	

Table 2: Features for the shift-reduce parser (SRP) and the graph-based maximum spanning tree (MST) parser. The √\* features are extracted from the heads of  $L_1$ ,  $L_2$  and  $Q$  for SRP and from each node of the edge for MST.

only 40 instances of OVERLAP relations were annotated when neither INCLUDES nor IS\_INCLUDED label matched, for evaluation purposes all instances of these relations were merged into the temporally coarse OVERLAP relation. Thus, the total number of OVERLAP relations in the corpus grew from 40 to 258 annotations in total.

To evaluate the parsing models (SRP and MST) we proposed two baselines. Both are based on the assumption of linear temporal structures of narratives as the temporal ordering process that was evidenced by studies in human text rewriting (Hickmann, 2003). The proposed baselines are:

- **LinearSeq:** A model that assumes all events occur in the order they are written, adding links between each pair of adjacent events, and labeling all links with the relation BEFORE.
- **ClassifySeq:** A model that links each pair of adjacent events, but trains a pair-wise classifier to predict the relation label for each pair. The

classifier is a support vector machine trained using the same features as the MST parser. This is an approximation of prior work, where the pairs of events to classify with a temporal relation were given as an input to the system. (Note that Section 6.2 will show that for our corpus, applying the model only to adjacent pairs of events is quite competitive for just getting the basic unlabeled link structure right.)

The Shift-Reduce parser (SRP; Section 4.1) and the graph-based, maximum spanning tree parser (MST; Section 4.2) are compared to these baselines.

## 6.1 Evaluation Criteria and Metrics

Model performance was evaluated using standard evaluation criteria for parser evaluations:

**Unlabeled Attachment Score (UAS)** The fraction of events whose head events were correctly predicted. This measures whether the correct pairs of events were linked, but not if they were linked by the correct relations.

**Labeled Attachment Score (LAS)** The fraction of events whose head events were correctly predicted with the correct relations. This measures both whether the correct pairs of events were linked and whether their temporal ordering is correct.

**Tree Edit Distance** In addition to the UAS and LAS the *tree edit distance* score has been recently introduced for evaluating dependency structures (Tsarfaty et al., 2011). The tree edit distance score for a tree  $\pi$  is based on the following operations  $\lambda \in \Lambda : \Lambda = \{\text{DELETE}, \text{INSERT}, \text{RELABEL}\}$ :

- $\lambda = \text{DELETE}$  delete a non-root node  $v$  in  $\pi$  with parent  $u$ , making the children of  $v$  the children of  $u$ , inserted in the place of  $v$  as a subsequence in the left-to-right order of the children of  $u$ .
- $\lambda = \text{INSERT}$  insert a node  $v$  as a child of  $u$  in  $\pi$  making it the parent of a consecutive subsequence of the children of  $u$ .
- $\lambda = \text{RELABEL}$  change the label of node  $v$  in  $\pi$

Any two trees  $\pi_1$  and  $\pi_2$  can be turned one into another by a sequence of edit operations  $\{\lambda_1, \dots, \lambda_n\}$ .

	UAS	LAS	UTEDS	LTEDS
LinearSeq	0.830	0.581	0.689	0.549
ClassifySeq	0.830	0.581	0.689	0.549
MST	0.837	0.614*	0.710	0.571
SRP	0.830	0.647*†	0.712	0.596*

Table 3: Performance levels of temporal structure parsing methods. A \* indicates that the model outperforms LinearSeq and ClassifiedSeq at  $p < 0.01$  and a † indicates that the model outperforms MST at  $p < 0.05$ .

Taking the shortest such sequence, the tree edit distance is calculated as the sum of the edit operation costs divided by the size of the tree (i.e. the number of words in the sentence). For temporal dependency trees, we assume each operation costs 1.0. The final score subtracts the edit distance from 1 so that a perfect tree has score 1.0. The *labeled* tree edit distance score (LTEDS) calculates sequences over the tree with all its labeled temporal relations, while the *unlabeled* tree edit distance score (UTEDS) treats all edges as if they had the same label.

## 6.2 Results

Table 3 shows the results of the evaluation. The unlabeled attachment score for the LinearSeq baseline was 0.830, suggesting that annotators were most often linking adjacent events. At the same time, the labeled attachment score was 0.581, indicating that even in fables, the stories are not simply linear, that is, there are many relations other than BEFORE. The ClassifySeq baseline performs identically to the LinearSeq baseline, which shows that the simple pairwise classifier was unable to learn anything beyond predicting all relations as BEFORE.

In terms of labeled attachment score, both dependency parsing models outperformed the baseline models – the maximum spanning tree parser achieved 0.614 LAS, and the shift-reduce parser achieved 0.647 LAS. The shift-reduce parser also outperformed the baseline models in terms of labeled tree edit distance, achieving 0.596 LTEDS vs. the baseline 0.549 LTEDS. These results indicate that dependency parsing models are a good fit to our whole-story timeline extraction task.

Finally, in comparing the two different dependency parsing models, we observe that the shift-reduce parser outperforms the maximum spanning

Error Type	Num.	%
OVERLAP → BEFORE	24	43.7
Attach to further head	18	32.7
Attach to nearer head	6	11.0
Other types of errors	7	12.6
Total	55	100

Table 4: Error distribution from the analysis of 55 errors of the Shift-Reduce parsing model.

tree parser in terms of labeled attachment score (0.647 vs. 0.614). It has been argued that graph-based models like the maximum spanning tree parser should be able to produce more globally consistent and correct dependency trees, yet we do not observe that here. A likely explanation for this phenomenon is that the shift-reduce parsing model allows for features describing previous parse decisions (similar to the incremental nature of human parse decisions), while the joint nature of the maximum spanning tree parser does not.

## 6.3 Error Analysis

To better understand the errors our model is still making, we examined two folds (55 errors in total in 20% of the evaluation data) and identified the major categories of errors:

- **OVERLAP → BEFORE:** The model predicts the correct head, but predicts its label as BEFORE, while the correct label is OVERLAP.
- **Attach to further head:** The model predicts the wrong head, and predicts as the head an event that is further away than the true head.
- **Attach to nearer head:** The model predicts the wrong head, and predicts as the head an event that is closer than the true head.

Table 4 shows the distribution of the errors over these categories. The two most common types of errors, OVERLAP → BEFORE and *Attach to further head*, account for 76.4% of all the errors.

The most common type of error is predicting a BEFORE relation when the correct answer is an OVERLAP relation. Figure 3 shows an example of such an error, where the model predicts that the Spendthrift *stood* before he *saw*, while the annotator indicates that the seeing happened during the

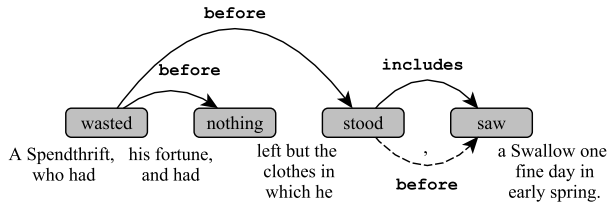


Figure 3: An OVERLAP  $\rightarrow$  BEFORE parser error. True links are solid lines; the parser error is the dotted line.

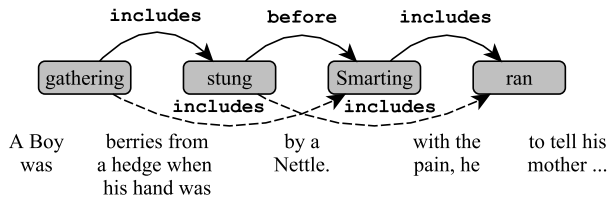


Figure 4: Parser errors attaching to further away heads. True links are solid lines; parser errors are dotted lines.

time in which he was standing. An analysis of these OVERLAP  $\rightarrow$  BEFORE errors suggests that they occur in scenarios like this one, where the duration of one event is significantly longer than the duration of another, but there are no direct cues for these duration differences. We also observe these types of errors when one event has many sub-events, and therefore the duration of the main event typically includes the durations of all the sub-events. It might be possible to address these kinds of errors by incorporating automatically extracted event duration information (Pan et al., 2006; Gusev et al., 2011).

The second most common error type of the model is the prediction of a head event that is further away than the head identified by the annotators. Figure 4 gives an example of such an error, where the model predicts that the *gathering* includes the *smarting*, instead of that the *gathering* includes the *stung*. The second error in the figure is also of the same type. In 65% of the cases where this type of error occurs, it occurs after the parser had already made a label classification error such as BEFORE  $\rightarrow$  OVERLAP. So these errors may be in part due to the sequential nature of shift-reduce parsing, where early errors propagate and cause later errors.

## 7 Discussion and Conclusions

In this article, we have presented an approach to temporal information extraction that represents the time-

line of a story as a temporal dependency tree. We have constructed an evaluation corpus where such temporal dependencies have been annotated over a set of 100 children’s stories. We have introduced two dependency parsing techniques for extracting story timelines and have shown that both outperform a rule-based baseline and a prior-work-inspired pair-wise classification baseline. Comparing the two dependency parsing models, we have found that a shift-reduce parser, which more closely mirrors the incremental processing of our human annotators, outperforms a graph-based maximum spanning tree parser. Our error analysis of the shift-reduce parser revealed that being able to estimate differences in event durations may play a key role in improving parse quality.

We have focused on children’s stories in this study, in part because they typically have simpler temporal structures (though not so simple that our rule-based baseline could parse them accurately). In most of our fables, there were only one or two characters with at most one or two simultaneous sequences of actions. In other domains, the timeline of a text is likely to be more complex. For example, in clinical records, descriptions of patients may jump back and forth between the patient history, the current examination, and procedures that have not yet happened.

In future work, we plan to investigate how to best apply the dependency structure approach to such domains. One approach might be to first group events into their *narrative containers* (Pustejovsky and Stubbs, 2011), for example, grouping together all events linked to the time of a patient’s examination. Then within each narrative container, our dependency parsing approach could be applied. Another approach might be to join the individual timeline trees into a document-wide tree via discourse relations or relations to the document creation time. Work on how humans incrementally process such timelines in text may help to decide which of these approaches holds the most promise.

## Acknowledgements

We would like to thank the anonymous reviewers for their constructive comments. This research was partially funded by the TERENCE project (EU FP7-257410) and the PARIS project (IWT SBO 110067).

## References

- [Amigó et al.2011] Enrique Amigó, Javier Artiles, Qi Li, and Heng Ji. 2011. An evaluation framework for aggregated temporal information extraction. In *SIGIR-2011 Workshop on Entity-Oriented Search*.
- [Artiles et al.2011] Javier Artiles, Qi Li, Taylor Cassidy, Suzanne Tamang, and Heng Ji. 2011. CUNY\_BLENDER TAC-KBP2011 temporal slot filling system description. In *Text Analytics Conference (TAC2011)*.
- [Bethard and Martin2007] Steven Bethard and James H. Martin. 2007. CU-TMP: Temporal relation classification using syntactic and semantic features. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 129–132, Prague, Czech Republic, June. ACL.
- [Bethard et al.2007] Steven Bethard, James H. Martin, and Sara Klingenstein. 2007. Finding temporal structure in text: Machine learning of syntactic temporal relations. *International Journal of Semantic Computing (IJSC)*, 1(4):441–458, 12.
- [Bethard et al.2012] Steven Bethard, Oleksandr Kolomiyets, and Marie-Francine Moens. 2012. Annotating narrative timelines as temporal dependency structures. In *Proceedings of the International Conference on Linguistic Resources and Evaluation*, Istanbul, Turkey, May. ELRA.
- [Boguraev and Ando2005] Branimir Boguraev and Rie Kubota Ando. 2005. TimeBank-driven TimeML analysis. In *Annotating, Extracting and Reasoning about Time and Events*. Springer.
- [Bramsen et al.2006] P. Bramsen, P. Deshpande, Y.K. Lee, and R. Barzilay. 2006. Inducing temporal graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 189–198. ACL.
- [Brewer and Lichtenstein1982] William F. Brewer and Edward H. Lichtenstein. 1982. Stories are to entertain: A structural-affect theory of stories. *Journal of Pragmatics*, 6(5-6):473 – 486.
- [Chambers and Jurafsky2008] N. Chambers and D. Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 698–706. ACL.
- [Cheng et al.2007] Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2007. NAIST.Japan: Temporal relation identification using dependency parsed tree. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 245–248, Prague, Czech Republic, June. ACL.
- [Chu and Liu1965] Y. J. Chu and T.H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*, pages 1396–1400.
- [Covington2001] M.A. Covington. 2001. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102.
- [Crammer and Singer2003] K. Crammer and Y. Singer. 2003. Ultraconservative online algorithms for multi-class problems. *Journal of Machine Learning Research*, 3:951–991.
- [Crammer et al.2006] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.
- [Edmonds1967] J. Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards*, pages 233–240.
- [Georgiadis2003] L. Georgiadis. 2003. Arborescence optimization problems solvable by Edmonds’ algorithm. *Theoretical Computer Science*, 301(1-3):427–437.
- [Gupta and Ji2009] Prashant Gupta and Heng Ji. 2009. Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort ’09, pages 369–372, Stroudsburg, PA, USA. ACL.
- [Gusev et al.2011] Andrey Gusev, Nathanael Chambers, Divye Raj Khilnani, Pranav Khaitan, Steven Bethard, and Dan Jurafsky. 2011. Using query patterns to learn the duration of events. In *Proceedings of the International Conference on Computational Semantics*, pages 145–154.
- [Hayes and Krippendorff2007] A.F. Hayes and K. Krippendorff. 2007. Answering the call for a standard reliability measure for coding data. *Communication Methods and Measures*, 1(1):77–89.
- [Hickmann2003] Maya Hickmann. 2003. *Children’s Discourse: Person, Space and Time Across Languages*. Cambridge University Press, Cambridge, UK.
- [Johnson-Laird1980] P.N. Johnson-Laird. 1980. Mental models in cognitive science. *Cognitive Science*, 4(1):71–115.
- [Krippendorff2004] K. Krippendorff. 2004. *Content analysis: An introduction to its methodology*. Sage Publications, Inc.
- [Linguistic Data Consortium2005] Linguistic Data Consortium. 2005. ACE (Automatic Content Extraction) English annotation guidelines for events version 5.4.3 2005.07.01.
- [Llorens et al.2010] Hector Llorens, Estela Saquete, and Borja Navarro. 2010. TIPSem (English and Spanish): Evaluating CRFs and semantic roles in TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 284–291, Uppsala, Sweden, July. ACL.

- [McDonald et al.2005] R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530. ACL.
- [McIntyre and Lapata2009] N. McIntyre and M. Lapata. 2009. Learning to tell tales: A data-driven approach to story generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 217–225. ACL.
- [Nivre2008] J. Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.
- [Pan et al.2006] Feng Pan, Rutu Mulkar, and Jerry R. Hobbs. 2006. Learning event durations from event descriptions. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 393–400, Sydney, Australia, July. ACL.
- [Pustejovsky and Stubbs2011] J. Pustejovsky and A. Stubbs. 2011. Increasing informativeness in temporal annotation. In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 152–160. ACL.
- [Pustejovsky et al.2003a] James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003a. TimeML: Robust specification of event and temporal expressions in text. In *Proceedings of the Fifth International Workshop on Computational Semantics (IWCS-5)*, Tilburg.
- [Pustejovsky et al.2003b] James Pustejovsky, Patrick Hanks, Roser Saurí, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, and Marcia Lazo. 2003b. The TimeBank corpus. In *Proceedings of Corpus Linguistics*, pages 647–656.
- [Tsarfaty et al.2011] R. Tsarfaty, J. Nivre, and E. Andersson. 2011. Evaluating dependency parsing: Robust and heuristics-free cross-annotation evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 385–396. ACL.
- [UzZaman and Allen2010] Naushad UzZaman and James Allen. 2010. TRIPS and TRIOS system for TempEval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283, Uppsala, Sweden, July. ACL.
- [Verhagen et al.2007] Marc Verhagen, Robert Gaizauskas, Frank Schilder, Graham Katz, and James Pustejovsky. 2007. SemEval2007 Task 15: TempEval temporal relation identification. In *SemEval-2007: 4th International Workshop on Semantic Evaluations*.
- [Verhagen et al.2010] Marc Verhagen, Roser Saurí, Tommaso Caselli, and James Pustejovsky. 2010. SemEval-2010 Task 13: TempEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10*, pages 57–62, Stroudsburg, PA, USA. ACL.
- [Yamada and Matsumoto2003] H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT*.
- [Yoshikawa et al.2009] K. Yoshikawa, S. Riedel, M. Asahara, and Y. Matsumoto. 2009. Jointly identifying temporal relations with Markov Logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 405–413. ACL.

# Labeling Documents with Timestamps: Learning from their Time Expressions

Nathanael Chambers

Department of Computer Science  
United States Naval Academy  
nchamber@usna.edu

## Abstract

Temporal reasoners for document understanding typically assume that a document's creation date is known. Algorithms to ground relative time expressions and order events often rely on this timestamp to assist the learner. Unfortunately, the timestamp is not always known, particularly on the Web. This paper addresses the task of automatic document timestamping, presenting two new models that incorporate rich linguistic features about time. The first is a discriminative classifier with new features extracted from the text's time expressions (e.g., 'since 1999'). This model alone improves on previous generative models by 77%. The second model learns probabilistic constraints between time expressions and the *unknown* document time. Imposing these learned constraints on the discriminative model further improves its accuracy. Finally, we present a new experiment design that facilitates easier comparison by future work.

## 1 Introduction

This paper addresses a relatively new task in the NLP community: automatic document dating. Given a document with unknown origins, what characteristics of its text indicate the year in which the document was written? This paper proposes a learning approach that builds constraints from a document's use of time expressions, and combines them with a new discriminative classifier that greatly improves previous work.

The temporal reasoning community has long depended on document timestamps to ground rela-

tive time expressions and events (Mani and Wilson, 2000; Llidó et al., 2001). For instance, consider the following passage from the TimeBank corpus (Pustejovsky et al., 2003):

And while there was no profit *this year* from discontinued operations, *last year* they contributed 34 million, before tax.

Reconstructing the timeline of events from this document requires extensive temporal knowledge, most notably, the document's creation date to ground its relative expressions (e.g., *this year* = 2012). Not only did the latest TempEval competitions (Verhagen et al., 2007; Verhagen et al., 2009) include tasks to link events to the (known) document creation time, but state-of-the-art event-event ordering algorithms also rely on these timestamps (Chambers and Jurafsky, 2008; Yoshikawa et al., 2009). This knowledge is assumed to be available, but unfortunately this is not often the case, particularly on the Web.

Document timestamps are growing in importance to the information retrieval (IR) and management communities as well. Several IR applications depend on knowledge of when documents were posted, such as computing document relevance (Li and Croft, 2003; Dakka et al., 2008) and labeling search queries with temporal profiles (Diaz and Jones, 2004; Zhang et al., 2009). Dating documents is similarly important to processing historical and heritage collections of text. Some of the early work that motivates this paper arose from the goal of automatically grounding documents in their historical contexts (de Jong et al., 2005; Kanhabua and Norvag, 2008; Kumar et al., 2011). This paper builds on their work

by incorporating more linguistic knowledge and explicit reasoning into the learner.

The first part of this paper describes a novel learning approach to document dating, presenting a discriminative model and rich linguistic features that have not been applied to document dating. Further, we introduce new features specific to absolute time expressions. Our model outperforms the generative models of previous work by 77%.

The second half of this paper describes a novel learning algorithm that orders time expressions against the unknown timestamp. For instance, the phrase *the second quarter of 1999* might be labeled as being *before* the timestamp. These labels impose constraints on the possible timestamp and narrow down its range of valid dates. We combine these constraints with our discriminative learner and see another relative improvement in accuracy by 9%.

## 2 Previous Work

Most work on dating documents has come from the IR and knowledge management communities interested in dating documents with unknown origins. de Jong et al. (2005) was among the first to automatically label documents with dates. They learned unigram language models (LMs) for specific time periods and scored articles with log-likelihood ratio scores. Kanhabua and Norvag (2008; 2009) extended this approach with the same model, but expanded its unigrams with POS tags, collocations, and tf-idf scores. They also integrated search engine results as features, but did not see an improvement. Both works evaluated on the news genre.

Recent work by Kumar et al. (2011) focused on dating Gutenberg short stories. As above, they learned unigram LMs, but instead measured the KL-divergence between a document and a time period’s LM. Our proposed models differ from this work by applying rich linguistic features, discriminative models, and by focusing on how time expressions improve accuracy. We also study the news genre.

The only work we are aware of within the NLP community is that of Dalli and Wilks (2006). They computed probability distributions over different time periods (e.g., months and years) for each observed token. The work is similar to the above IR work in its bag of words approach to classification.

They focused on finding words that show periodic spikes (defined by the word’s standard deviation in its distribution over time), weighted with inverse document frequency scores. They evaluated on a subset of the Gigaword Corpus (Graff, 2002).

The experimental setup in the above work (except Kumar et al. who focus on fiction) all train on news articles from a particular time period, and test on articles in the same time period. This leads to possible overlap of training and testing data, particularly since news is often reprinted across agencies the same day. In fact, one of the systems in Kanhabua and Norvag (2008) simply searches for one training document that best matches a test document, and assigns its timestamp. We intentionally deviate from this experimental design and instead create temporally disjoint train/test sets (see Section 5).

Finally, we extend this previous work by focusing on aspects of language not yet addressed for document dating: linguistic structure and absolute time expressions. The majority of articles in our dataset contain time expressions (e.g., the year 1998), yet these have not been incorporated into the models despite their obvious connection to the article’s timestamp. This paper first describes how to include time expressions as traditional features, and then describes a more sophisticated temporal reasoning component that naturally fits into our classifier.

## 3 Timestamp Classifiers

Labeling documents with timestamps is similar to topic classification, but instead of choosing from topics, we choose the most likely year (or other granularity) in which it was written. We thus begin with a bag-of-words approach, reproducing the generative model used by both de Jong (2005) and Kanhabua and Norvag (2008; 2009). The subsequent sections then introduce our novel classifiers and temporal reasoners to compare against this model.

### 3.1 Language Models

The model of de Jong et al. (2005) uses the normalized log-likelihood ratio (NLLR) to score documents. It weights tokens by the ratio of their probability in a specific year to their probability over the entire corpus. The model thus requires an LM for each year and an LM for the entire corpus:

$$NLLR(D, Y) = \sum_{w \in D} P(w|D) * \log\left(\frac{P(w|Y)}{P(w|C)}\right) \quad (1)$$

where  $D$  is the target document,  $Y$  is the time span (e.g., a year), and  $C$  is the distribution of words in the corpus across all years. A document is labeled with the year that satisfies  $\operatorname{argmax}_Y NLLR(D, Y)$ . They adapted this model from earlier work in the IR community (Kraaij, 2004). We apply Dirichlet-smoothing to the language models (as in de Jong et al.), although the exact choice of  $\alpha$  did not significantly alter the results, most likely due to the large size of our training corpus. Kanhabua and Norvag added an entropy factor to the summation, but we did not see an improvement in our experiments.

The unigrams  $w$  are lowercased tokens. We will refer to this de Jong et al. model as the **Unigram NLLR**. Follow-up work by Kanhabua and Norvag (2008) applied two filtering techniques to the unigrams in the model:

1. **Word Classes**: include only nouns, verbs, and adjectives as labeled by a POS tagger
2. **IDF Filter**: include only the top-ranked terms by tf-idf score

We also tested with these filters, choosing a cutoff for the top-ranked terms that optimized performance on our development data. We also stemmed the words as Kanhabua and Norvag suggest. This model is the **Filtered NLLR**.

Kanhabua and Norvag also explored what they termed *collocation features*, but lacking details on how collocations were included (or learned), we could not reproduce this for comparison. However, we instead propose using NER labels to extract what may have counted as collocations in their data. Named entities are important to document dating due to the nature of people and places coming in and out of the news at precise moments in time. We compare the NER features against the Unigram and Filtered NLLR models in our final experiments.

### 3.2 Discriminative Models

In addition to reproducing the models from previous work, we also trained a new discriminative version with the same features. We used a MaxEnt model and evaluated with the same filtering methods based

on POS tags and tf-idf scores. The model performed best on the development data without any filtering or stemming. The final results (Section 6) only use the lowercased unigrams. Ultimately, this MaxEnt model vastly outperforms these NLLR models.

### 3.3 Models with Time Expressions

The above language modeling and MaxEnt approaches are token-based classifiers that one could apply to any topic classification domain. Barring other knowledge, the learners solely rely on the observed frequencies of unigrams in order to decide which class is most likely. However, document dating is not just a simple topic classification application, but rather relates to temporal phenomena that is often explicitly described in the text itself. Language contains words and phrases that discuss the very time periods we aim to recover. These expressions should be better incorporated into the learner.

#### 3.3.1 Motivation

Let the following snippet serve as a text example with an ambiguous creation time:

Then there's the fund-raiser at the American Museum of Natural History, which plans to welcome about 1,500 guests paying \$1,000 to \$5,000. Their tickets will entitle them to a preview of...the new Hayden Planetarium.

Without extremely detailed knowledge about the American Museum of Natural History, the events discussed here are difficult to place in time, let alone when the author reported it. However, time expressions are sometimes included, and the last sentence in the original text contains a helpful relative clause:

Their tickets will entitle them to a preview of...the new Hayden Planetarium, *which does not officially open until February 2000.*

This one clause is more valuable than the rest of the document, allowing us to infer that the document's timestamp is *before* February, 2000. An educated guess might surmise the article appeared in the year prior, 1999, which is the correct year. At the very least, this clause should eliminate all years after 2000 from consideration. Previous work on document dating does not integrate this information except to include the unigram '2000' in the model.



This paper discusses two complementary ways to learn and reason about this information. The first is to simply add richer time-based features into the model. The second is to build separate learners that can assign probabilities to entire ranges of dates, such as *all years following 2000* in the example above. We begin with the feature-based model.

### 3.3.2 Time Features

To our knowledge, the following time features have not been used in a document dating setting. We use the freely available Stanford Parser and NER system<sup>1</sup> to generate the syntactic interpretation for these features. We then train a MaxEnt classifier and compare against previous work.

**Typed Dependency:** The most basic time feature is including governors of year mentions and the relation between them. This covers important contexts that determine the semantics of the time frame, like prepositions. For example, consider the following context for the mention *1997*:

Torre, who watched the Kansas City Royals beat the Yankees, 13-6, on Friday for the first time since 1997.

The resulting feature is ‘*since pobj 1997*’.

**Typed Dependency POS:** Similar to Typed Dependency, this feature uses POS tags of the dependency relation’s governor. The feature from the previous example is now ‘*PP pobj 1997*’. This generalizes the features to capture time expressions with prepositions, as noun modifiers, or other constructs.

**Verb Tense:** An important syntactic feature for temporal positioning is the tense of the verb that dominates the time expression. A past tense verb situates the phrase *in 2003* differently than one in the future. We traverse the sentence’s parse tree until a governor with a VB\* tag is found, and determine its tense through hand constructed rules based on the structure of the parent VP. The verb tense feature takes a value of *past*, *present*, *future*, or *undetermined*.

**Verb Path:** The verb path feature is the dependency path from the nearest verb to the year expression. The following snippet will include the feature, ‘*expected prep in pobj 2002*’.

<sup>1</sup><http://nlp.stanford.edu/software>

### Finance Article from Jan. 2002

Text Snippet	Relation to 2002
...started a hedge fund before the market peaked in 2000.	before
The peak in economic activity was the 4th quarter of 1999.	before
...might have difficulty in the latter part of 2002.	simultaneous

Figure 1: Three year mentions and their relation to the document creation year. Relations can be correctly identified for training using known document timestamps.

Supervising them is Vice President Hu Jintao, who appears to be Jiang’s favored successor if he retires from leadership as expected in 2002.

**Named Entities:** Although not directly related to time expressions, we also include n-grams of tokens that are labeled by an NER system using Person, Organization, or Location. People and places are often discussed during specific time periods, particularly in the news genre. Collecting named entity mentions will differentiate between an article discussing a *bill* and one discussing the US President, *Bill Clinton*. We extract NER features as sequences of uninterrupted tokens labeled with the same NER tag, ignoring unigrams (since unigrams are already included in the base model). Using the Verb Path example above, the bigram feature *Hu Jintao* is included.

## 4 Learning Time Constraints

This section departs from the above *document* classifiers and instead classifies individual emphyyear mentions. The goal is to automatically learn **temporal constraints** on the document’s timestamp.

Instead of predicting a single year for a document, a temporal constraint predicts a range of years. Each time mention, such as ‘*not since 2009*’, is a constraint representing its relation to the document’s timestamp. For example, the mentioned year ‘2009’ must occur *before* the year of document creation. This section builds a classifier to label time mentions with their relations (e.g., before, after, or simultaneous with the document’s timestamp), enabling these mentions to constrain the document classifiers described above. Figure 1 gives an example of time mentions and the desired labels we wish to learn.

To better motivate the need for constraints, let

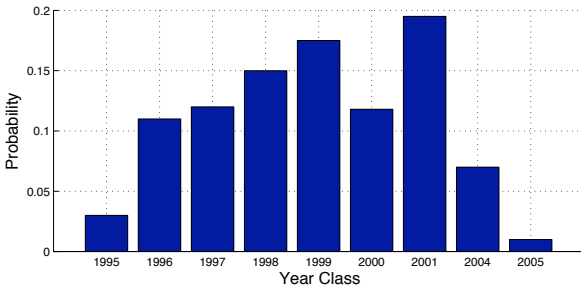


Figure 2: Distribution over years for a single document as output by a MaxEnt classifier.

Figure 2 illustrate a typical distribution output by a document classifier for a training document. Two of the years appear likely (1999 and 2001), however, the document contains a time expression that seems to impose a strict constraint that should eliminate 2001 from consideration:

Their tickets will entitle them to a preview of...the new Hayden Planetarium, which does not officially open until February 2000.

The clause *until February 2000* in a present tense context may not definitively identify the document’s timestamp (1999 is a good guess), but as discussed earlier, it should remove all future years beyond 2000 from consideration. We thus want to impose a constraint based on this phrase that says, loosely, ‘this document was likely written before 2000’.

The document classifiers described in previous sections cannot capture such ordering information. Our new time features in Section 3.3.2 add richer time information (such as *until pobj 2000* and *open prep until pobj 2000*), but they compete with many other features that can mislead the final classification. An independent constraint learner may push the document classifier in the right direction.

#### 4.1 Constraint Types

We learn several types of constraints between each year mention and the document’s timestamp. Year mentions are defined as tokens with exactly four digits, numerically between 1900 and 2100. Let  $T$  be the document timestamp’s year, and  $M$  the year mention. We define three **core relations**:

1. **Before** Timestamp:  $M < T$
2. **After** Timestamp:  $M > T$
3. **Same** as Timestamp:  $M == T$

We also experiment with 7 **fine-grained relations**:

1. **One year Before** Timestamp:  $M == T - 1$
2. **Two years Before** Timestamp:  $M == T - 2$
3. **Three+ years Before** Timestamp:  $M < T - 2$
4. **One year After** Timestamp:  $M == T + 1$
5. **Two years After** Timestamp:  $M == T + 2$
6. **Three+ years After** Timestamp:  $M > T + 2$
7. **Same Year and Timestamp**:  $M == T$

Obviously the more fine-grained a relation, the better it can inform a classifier. We experiment with these two granularities to compare performance.

The learning process is a typical training environment where year mentions are treated as labeled training examples. Labels for year mentions are automatically computed by comparing the actual timestamp of the training document (all documents in Gigaword have dates) with the integer value of the year token. For example, a document written in 1997 might contain the phrase, “in the year 2000”. The year token (2000) is thus *three+ years after* the timestamp (1997). We use this relation for the year mention as a labeled training example.

Ultimately, we want to use similar syntactic constructs in training so that “in the year 2000” and “in the year 2003” mutually inform each other. We thus compute the label for each time expression, and replace the integer year with the generic *YEAR* token to generalize mentions. The text for this example becomes “in the year *YEAR*” (labeled as *three+ years after*). We train a MaxEnt model on each year mention, to be described next. Table 2 gives the overall counts for the core relations in our training data. The vast majority of year mentions are references to the future (e.g. after the timestamp).

#### 4.2 Constraint Learner

The features we use to classify year mentions are given in Table 1. The same time features in the document classifier of Section 3.3.2 are included, as well as several others specific to this constraint task.

We use a MaxEnt classifier trained on the individual year mentions. Documents often contain multiple (and different) year mentions; all are included in training and testing. This classifier labels mentions with relations, but in order to influence the document classifier, we need to map the relations to individual

### Time Constraint Features

Typed Dep.	Same as Section 3.3.2
Verb Tense	Same as Section 3.3.2
Verb Path	Same as Section 3.3.2
Decade	The decade of the year mention
Bag of Words	Unigrams in the year’s sentence
n-gram	The 4-gram and 3-gram that end with the year
n-gram POS	The 4-gram and 3-gram of POS tags that end with the year

Table 1: Features used to classify year expressions.

Constraint	Count
After Timestamp	1,203,010
Before Timestamp	168,185
Same as Timestamp	141,201

Table 2: Training size of year mentions (and their relation to the document timestamp) in Gigaword’s NYT section.

year predictions. Let  $T_d$  be the set of mentions in document  $d$ . We represent a MaxEnt classifier by  $P_Y(R|t)$  for a time mention  $t \in T_d$  and possible relations  $R$ . We map this distribution over relations to a distribution over years by defining  $P_{year}(Y|d)$ :

$$P_{year}(y|d) = \frac{1}{Z(T_d)} \sum_{t \in T_d} P_Y(\text{rel}(\text{val}(t) - y)|t) \quad (2)$$

$$\text{rel}(x) = \begin{cases} \textit{before} & \text{if } x < 0 \\ \textit{after} & \text{if } x > 0 \\ \textit{simultaneous} & \text{otherwise} \end{cases} \quad (3)$$

where  $\text{val}(t)$  is the integer year of the year mention and  $Z(T_d)$  is the partition function. The  $\text{rel}(\text{val}(t) - y)$  function simply determines if the year mention  $t$  (e.g., 2003) is before, after, or overlaps the year we are predicting for the document’s unknown timestamp  $y$ . We use a similar function for the seven fine-grained relations. Figure 3 visually illustrates how  $P_{year}(y|d)$  is constructed from three year mentions.

### 4.3 Joint Classifier

Finally, given the document classifiers of Section 3 and the constraint classifier just defined in Section 4, we create a joint model combining the two with the following linear interpolation:

$$P(y|d) = \lambda P_{doc}(y|d) + (1 - \lambda) P_{year}(y|d) \quad (4)$$

where  $y$  is a year, and  $d$  is the document.  $\lambda$  was set to 0.35 by maximizing accuracy on the dev set. See

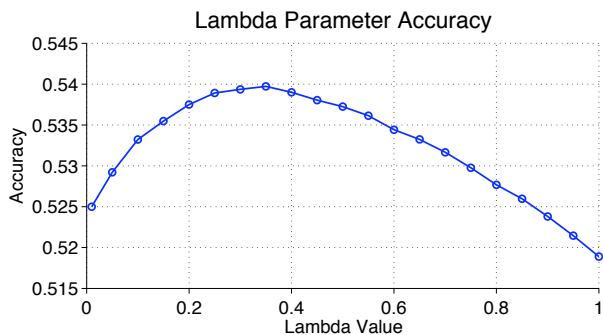


Figure 4: Development set accuracy and  $\lambda$  values.

Figure 4. This optimal  $\lambda = .35$  weights the constraint classifier higher than the document classifier.

## 5 Datasets

This paper uses the New York Times section of the Gigaword Corpus (Graff, 2002) for evaluation. Most previous work on document dating evaluates on the news genre, so we maintain the pattern for consistency. Unfortunately, we cannot compare to these previous experiments because of differing evaluation setups. Dalli and Wilks (2006) is most similar in their use of Gigaword, but they chose a random set of documents that cannot be reproduced. We instead define specific segments of the corpus for evaluation.

The main goal for this experiment setup was to establish specific training, development, and test sets. One of the potential difficulties in testing with news articles is that the same story is often reprinted with very minimal (or no) changes. Over 10% of the documents in the New York Times section of the Gigaword Corpus are exact or approximate duplicates of another document in the corpus<sup>2</sup>. A training set for document dating must not include duplicates from the test set.

We adopt the intuition behind the experimental setup used in other NLP domains, like parsing, where the entire test set is from a contiguous section of the corpus (as opposed to randomly selected examples across the corpus). As the parsing community trains on sections 2-21 of the Penn Treebank (Marcus et al., 1993) and tests on section 23, we create Gigaword sections by isolating specific months.

<sup>2</sup>*Approximate duplicate* is defined as an article whose first two sentences exactly match the first two of another article. Only the second matched document is counted as a duplicate.

### Year Distributions for Three Time Expressions

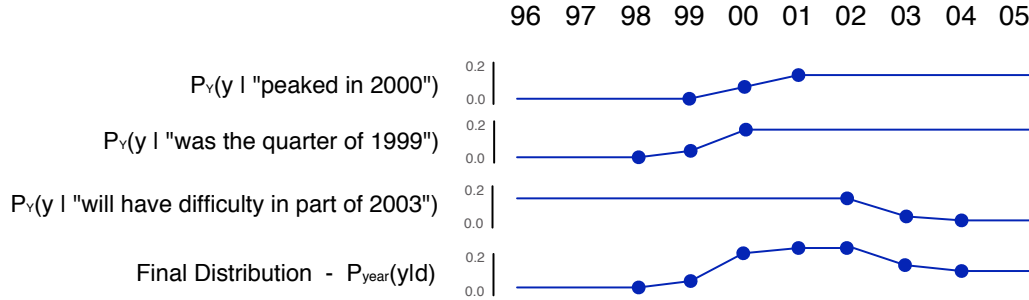


Figure 3: Three year mentions in a document and the distributions output by the learner. The document is from 2002. The dots indicate the *before*, *same*, and *after* relation probabilities. The combination of three constraints results in a final distribution that gives the years 2001 and 2002 the highest probability. This distribution can help a document classifier make a more informed final decision.

<b>Training</b>	Jan-May and Sep-Dec
<b>Development</b>	July
<b>Testing</b>	June and August

In other words, the development set includes documents from July 1995, July 1996, July 1997, etc. We chose the dev/test sets to be in the middle of the year so that the training set includes documents on both temporal sides of the test articles. We include years 1995-2001 and 2004-2006, but skip 2002 and 2003 due to their abnormally small size compared to the other years.

Finally, we experiment in a balanced data setting, training and testing on the same number of documents from each year. The test set includes 11,300 documents in each year (months June and August) for a total of 113,000 test documents. The development set includes 7,300 from July of each year. Training includes approximately 75,000 documents in each year with some years slightly less than 75,000 due to their smaller size in the corpus. The total number of training documents for the 10 evaluated years is 725,468. The full list of documents is online at [www.usna.edu/Users/cs/nchamber/data/timestamp](http://www.usna.edu/Users/cs/nchamber/data/timestamp).

## 6 Experiments and Results

We experiment on the Gigaword corpus as described in Section 5. Documents are tokenized and parsed with the Stanford Parser. The year in the timestamp is retrieved from the document's Gigaword ID which contains the year and day the article was re-

trieved. Year mentions are extracted from documents by matching all tokens with exactly four digits whose integer is in the range of 1900 and 2100.

The MaxEnt classifiers are also from the Stanford toolkit, and both the document and year mention classifiers use its default settings (quadratic prior). The  $\lambda$  factor in the joint classifier is optimized on the development set as described in Section 4.3. We also found that dev results improved when training ignores the border months of Jan, Feb, and Dec. The features described in this paper were selected solely by studying performance on the development set. The final reported results come from running on the test set once at the end of this study.

Table 3 shows the results on the Test set for all document classifiers. We measure accuracy to compare overall performance since the test set is a balanced set (each year has the same number of test documents). **Unigram NLLR** and **Filtered NLLR** are the language model implementations of previous work as described in Section 3.1. **MaxEnt Unigram** is our new discriminative model for this task. **MaxEnt Time** is the discriminative model with rich time features (but not NER) as described in Section 3.3.2 (**Time+NER** includes NER). Finally, the **Joint** model is the combined document and year mention classifiers as described in Section 4.3. Table 4 shows the F1 scores of the Joint model by year.

Our new MaxEnt model outperforms previous work by 55% relative accuracy. Incorporating time features further improves the relative accuracy by

Model	Overall Accuracy
Random Guess	10.0%
Unigram NLLR	24.1%
Filtered NLLR	29.1%
MaxEnt Unigram	45.1%
MaxEnt Time	48.3%
MaxEnt Time+NER	51.4%
Joint	53.4%

Table 3: Performance as measured by accuracy. The predicted year must exactly match the actual year.

	95	96	97	98	99	00	01	02
<b>P</b>	.57	.49	.52	.48	.47	.51	.51	.59
<b>R</b>	.54	.56	.62	.44	.48	.48	.46	.57
<b>F1</b>	.55	.52	.57	.46	.48	.49	.48	.58

Table 4: Yearly results for the Joint model. 2005/06 are omitted due to space, with F1 .56 and .63, respectively.

7%, and adding NER by another 6%. Total relative improvement in accuracy is thus almost 77% from the Time+NER model over Filtered NLLR. Further, the temporal constraint model increases this best classifier by another 3.9%. All improvements are statistically significant ( $p < 0.000001$ , McNemar’s test, 2-tailed). Table 6 shows that performance increased most on the documents that contain at least one year mention (60% of the corpus).

Finally, Table 5 shows the results of the temporal constraint classifiers on year mentions. Not surprisingly, the fine-grained performance is quite a bit lower than the core relations. The full Joint results in Table 3 use the three core relations, but the seven fine-grained relations give approximately the same results. Its lower accuracy is mitigated by the finer granularity (i.e., the majority class baseline is lower).

## 7 Discussion

The main contribution of this paper is the discriminative model (54% improvement) and a new set of

	<b>P</b>	<b>R</b>	<b>F1</b>
Before Timestamp	.95	.98	.96
Same as Timestamp	.73	.57	.64
After Timestamp	.84	.81	.82
Overall Accuracy	<b>92.2%</b>		
Fine-Grained Accuracy	<b>70.1%</b>		

Table 5: Precision, recall, and F1 for the core relations. Accuracy for both core and fine-grained.

	All	With Year Mentions
MaxEnt Unigram	45.1%	46.1%
MaxEnt Time+NER	51.4%	54.3%
Joint	53.4%	57.7%

Table 6: Accuracy on all documents and documents with at least one year mention (about 60% of the corpus).

features for document dating (14% improvement). Such a large performance boost makes clear that the log likelihood and entropy approaches from previous work are not as effective as discriminative models on a large training corpus. Further, token-based features do not capture the implicit references to time in language. Our richer syntax-based features only apply to year mentions, but this small textual phenomena leads to a surprising 13% relative improvement in accuracy. Table 6 shows that a significant chunk of this improvement comes from documents containing year mentions, as expected.

The year constraint learner also improved performance. Although most of its features are in the document classifier, by learning constraints it captures a different picture of time that a traditional document classifier does not address. Combining this picture with the document classifier leads to another 3.9% relative improvement. Although we focused on year mentions here, there are several avenues for future study, including explorations of how other types of time expressions might inform the task. These constraints might also have applications to the ordering tasks of recent TempEval competitions.

Finally, we presented a new evaluation setup for this task. Previous work depended on having training documents in the same week and day of the test documents. We argued that this may not be an appropriate assumption in some domains, and particularly problematic for the news genre. Our proposed evaluation setup instead separates training and testing data across months. The results show that log-likelihood ratio scores do not work as well in this environment. We hope our explicit train/test environment encourages future comparison and progress on document dating.

## Acknowledgments

Many thanks to Stephen Guo and Dan Jurafsky for early ideas and studies on this topic.

## References

- Nathanael Chambers and Dan Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)*, Hawaii, USA.
- W. Dakka, L. Gravano, and P. G. Ipeirotis. 2008. Answering general time sensitive queries. In *Proceedings of the 17th International ACM Conference on Information and Knowledge Management*, pages 1437–1438.
- Angelo Dalli and Yorick Wilks. 2006. Automatic dating of documents and temporal text classification. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 17–22.
- Franciska de Jong, Henning Rode, and Djoerd Hiemstra. 2005. Temporal language models for the disclosure of historical text. In *Humanities, computers and cultural heritage: Proceedings of the XVIth International Conference of the Association for History and Computing (AHC 2005)*.
- Fernando Diaz and Rosie Jones. 2004. Using temporal profiles of queries for precision prediction. In *Proceedings of the 27th Annual International ACM Special Interest Group on Information Retrieval Conference*.
- David Graff. 2002. English Gigaword. *Linguistic Data Consortium*.
- Nattiya Kanhabua and Kjetil Norvag. 2008. Improving temporal language models for determining time of non-timestamped documents. In *Proceedings of the 12th European conference on Research and Advanced Technology for Digital Libraries*.
- Nattiya Kanhabua and Kjetil Norvag. 2009. Using temporal language models for document dating. *Lecture Notes in Computer Science: machine learning and knowledge discovery in databases*, 5782.
- W. Kraaij. 2004. *Variations on language modeling for information retrieval*. Ph.D. thesis, University of Twente.
- Abhimanu Kumar, Matthew Lease, and Jason Baldridge. 2011. Supervised language modeling for temporal resolution of texts. In *Proceedings of CIKM*.
- Xiaoyan Li and W. Bruce Croft. 2003. Time-based language models. In *Proceedings of the twelfth international conference on Information and knowledge management*.
- Dolores M. Llidó, Rafael Llavori, and Mariá J. Aramburu. 2001. Extracting temporal references to assign document event-time periods. In *Proceedings of the 12th International Conference on Database and Expert Systems Applications*.
- Inderjeet Mani and George Wilson. 2000. Robust temporal processing of news. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19.
- James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, David Day, Lisa Ferro, Robert Gaizauskas, Marcia Lazo, Andrea Setzer, and Beth Sundheim. 2003. The timebank corpus. *Corpus Linguistics*, pages 647–656.
- Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Workshop on Semantic Evaluations*.
- Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Jessica Moszkowicz, and James Pustejovsky. 2009. The tempeval challenge: identifying temporal relations in text. *Special Issue: Computational Semantic Analysis of Language: SemEval-2007 and Beyond*, 43(2):161–179.
- Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with markov logic. In *Proceedings of the Association for Computational Linguistics (ACL)*.
- Ruiqiang Zhang, Yi Chang, Zhaohui Zheng, Donald Metzler, and Jian yun Nie. 2009. Search result re-ranking by feedback control adjustment for time-sensitive query. In *Proceedings of the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

# Temporally Anchored Relation Extraction

Guillermo Garrido, Anselmo Peñas, Bernardo Cabaleiro, and Álvaro Rodrigo

NLP & IR Group at UNED

Madrid, Spain

{ggarrido, anselmo, bcabaleiro, alvarory}@lsi.uned.es

## Abstract

Although much work on relation extraction has aimed at obtaining static facts, many of the target relations are actually *fluents*, as their validity is naturally anchored to a certain time period. This paper proposes a methodological approach to temporally anchored relation extraction. Our proposal performs distant supervised learning to extract a set of relations from a natural language corpus, and anchors each of them to an interval of temporal validity, aggregating evidence from documents supporting the relation. We use a rich graph-based document-level representation to generate novel features for this task. Results show that our implementation for temporal anchoring is able to achieve a 69% of the upper bound performance imposed by the relation extraction step. Compared to the state of the art, the overall system achieves the highest precision reported.

## 1 Introduction

A question that arises when extracting a relation is how to capture its temporal validity: Can we assign a period of time when the obtained relation held? As pointed out in (Ling and Weld, 2010), while much research in automatic relation extraction has focused on distilling static facts from text, many of the target relations are in fact *fluents*, dynamic relations whose truth value is dependent on time (Russell and Norvig, 2010).

The *Temporally anchored relation extraction* problem consists in, given a natural language text document corpus,  $C$ , a target entity,  $e$ , and a target

relation,  $r$ , extracting from the corpus the value of that relation for the entity, and a temporal interval for which the relation was valid.

In this paper, we introduce a methodological approach to temporal anchoring of relations automatically extracted from unrestricted text. Our system (see Figure 1) extracts relational facts from text using distant supervision (Mintz et al., 2009) and then anchors the relation to an interval of temporal validity. The intuition is that a distant supervised system can effectively extract relations from the source text collection, and a straightforward date aggregation can then be applied to anchor them. We propose a four step process for temporal anchoring: (1) represent temporal evidence; (2) select temporal information relevant to the relation; (3) decide how a relational fact and its relevant temporal information are themselves related; and (4) aggregate imprecise temporal intervals across multiple documents. In contrast with previous approaches that aim at intra-document temporal information extraction (Ling and Weld, 2010), we focus on mining a corpus aggregating temporal evidences across the supporting documents.

We address the following research questions: (1) Validate whether distant supervised learning is suitable for the task, and evaluate its shortcomings. (2) Explore whether the use of features extracted from a document-level rich representation could improve distant supervised learning. (3) Compare the use of document metadata against temporal expressions within the document for relation temporal anchoring. (4) Analyze how, in a pipeline architecture, the propagation of errors limits the overall system's

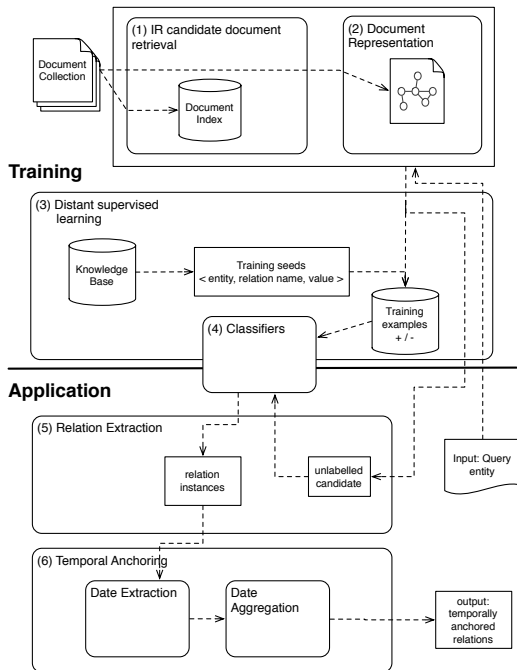


Figure 1: System overview diagram.

performance.

The representation we use for temporal information is detailed in section 2; the rich document-level representation we exploit is described in section 3. For a query entity and target relation, the system first performs relation extraction (section 4); then, we find and aggregate time constraint evidence for the same relation across different documents, to establish a temporal validity anchor interval (section 5). Empirical comparative evaluation of our approach is introduced in section 6; while some related work is shown in section 7 and conclusions in section 8.

## 2 Temporal Anchors

We will denominate *relation instance* a triple  $\langle \text{entity}, \text{relation name}, \text{value} \rangle$ . We aim at anchoring relation instances to their temporal validity. We need a representation flexible enough to capture the imprecise temporal information available in text, but expressed in a structured style. Allen’s (1983) interval-based algebra for temporal representation and reasoning, underlies much research, such as the Tempeval challenges (Verhagen et al., 2007; Pustejovsky and Verhagen, 2009). Our task is different, as we focus on obtaining the temporal interval associated to a fact, rather than reasoning about the

temporal relations among the events appearing in a single text.

Let us assume that each relation instance is valid during a certain temporal interval,  $I = [t_0, t_f]$ . This sharp temporal interval fails to capture the imprecision of temporal boundaries conveyed in natural language text. The Temporal Slot Filling task at TAC-KBP 2011 (Ji et al., 2011) proposed a 4-tuple representation that we will refer to as *imprecise anchor intervals*. An imprecise temporal interval is defined as an ordered 4-tuple of time points:  $(t_1, t_2, t_3, t_4)$ , with the following semantics: the relation is true for a period which starts at some point between  $t_1$  and  $t_2$  and ends between  $t_3$  and  $t_4$ . It should hold that:  $t_1 \leq t_2$ ,  $t_3 \leq t_4$ , and  $t_1 \leq t_4$ . Any of the four endpoints can be left unconstrained ( $t_1$  or  $t_3$  would be  $-\infty$ , and  $t_2$  or  $t_4$  would be  $+\infty$ ). This representation is flexible and expressive, although it cannot capture certain types of information (Ji et al., 2011).

## 3 Document Representation

We use a rich document representation that employs a graph structure obtained by augmenting the syntactic dependency analysis of the document with semantic information.

A document  $D$  is represented as a *document graph*  $G_D$ ; with node set  $V_D$  and edge set,  $E_D$ . Each node  $v \in V_D$  represents a *chunk* of text, which is a sequence of words<sup>1</sup>. Each node is labeled with a dictionary of attributes, some of which are common for every node: the words it contains, their part-of-speech annotations (POS) and lemmas. Also, a representative *descriptor*, which is a normalized string value, is generated from the chunks in the node. Certain nodes are also annotated with one or more *types*. There are three families of types: Events (verbs that describe an action, annotated with tense, polarity and aspect); standardized Time Expressions; and Named Entities, with additional annotations such as gender or age.

Edges in the document graph,  $e \in E_D$ , represent four kinds of relations between the nodes:

- Syntactic: a dependency relation.
- Coreference: indicates that two chunks refer to

<sup>1</sup>Most chunks consist in one word; we join words into a chunk (and a node) in two cases: a multi-word named entity and a verb and its auxiliaries.



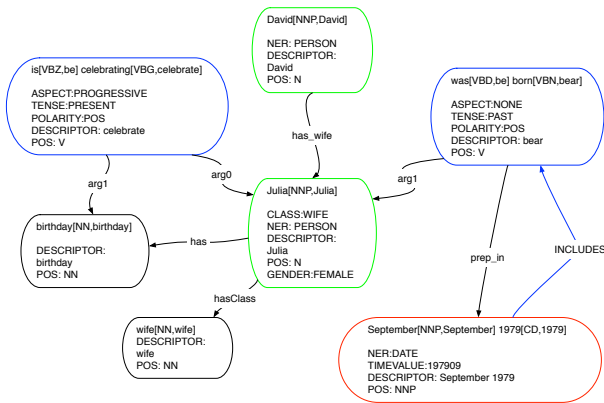


Figure 2: Collapsed document graph representation,  $G_C$ , for the sample text document “David’s wife, Julia, is celebrating her birthday. She was born in September 1979”.

the same *discourse referent*.

- Semantic relations between two nodes, such as `hasClass`, `hasProperty` and `hasAge`.
- Temporal relations between events and time expressions.

The processing includes dependency parsing, named entity recognition and coreference resolution, done with the Stanford CoreNLP software (Klein and Manning, 2003); and events and temporal information extraction, via the TARSQI Toolkit (Verhagen et al., 2005).

The document graph  $G_D$  is then further transformed into a *collapsed document graph*,  $G_C$ . Each node of  $G_C$  clusters together coreferent nodes, representing a *discourse referent*. Thus, a node  $u$  in  $G_C$  is a cluster of nodes  $u_1, \dots, u_k$  of  $G_D$ . There is an edge  $(u, v)$  in  $G_C$  if there was an edge between any of the nodes clustered into  $u$  and any of the nodes  $v_1, \dots, v_{k'}$ . The coreference edges do not appear in this representation. Additional semantic information is also blended into this representation: normalization of genitives, semantic class indicators inferred from appositions and genitives, and gender annotation inferred from pronouns. A final graph example can be seen in Figure 2.

## 4 Distant Supervised Relation Extraction

To perform relation extraction, our proposal follows a distant supervision approach (Mintz et al., 2009), which has also inspired other slot filling systems (Agirre et al., 2009; Surdeanu et al., 2010). We capture long distance relations by introducing

a document-level representation and deriving novel features from deep syntactic and semantic analysis.

**Seed harvesting.** From a reference Knowledge Base (KB), we extract a set of relation triples or *seeds*:  $\langle \text{entity}, \text{relation}, \text{value} \rangle$ , where the *relation* is one of the target relations. Our document-level distant supervision assumption is that if entity and value are found in a document graph (see section 3), and there is a path connecting them, then the document expresses the relation.

**Relation candidates gathering.** From a seed triple, we retrieve candidate documents that contain both the entity and value, within a span of 20 tokens, using a standard IR approach. Then, entity and value are matched to the document graph representation. We first use approximate string comparison to find nodes matching the seed entity. After an entity node has been found we use local breadth-first-search (BFS) to find a matching value and the shortest connecting path between them. We enforce the Named Entity type of entity and value to match a expected type, predefined for the relation.

Our procedure traverses the document graph looking for entity and value nodes meeting those conditions; when found, we generate features for a *positive example* for the relation<sup>2</sup>. If we encounter a node that matches the expected NE type of the relation, but does not match the seed value, we generate a *negative example* for that relation.

**Training.** From positive and negative examples, we generate binary features; some of them are inspired by previous work (Surdeanu and Ciaramita, 2007; Mintz et al., 2009; Riedel et al., 2010; Surdeanu et al., 2010), and others are novel, taking advantage of our graph representation. Table 1 summarizes our choice of features. Features appearing in less than 5 training examples were discarded.

**Relation instance extraction.** Given an input entity and a target relation, we aim at finding a filler value for a relation instance. This task is known as Slot Filling. From the set of retrieved documents relevant to the query entity, represented as document graphs,

<sup>2</sup>From the collapsed document graph representation we obtained an average of 9213 positive training examples per slot; from the uncollapsed document graph, a slightly lower average of 8178.5 positive examples per slot.

Feature name	Description
path	dependency path between ENTITY and VALUE in the sentence
$X$ -annotation	NE annotations for $X$
$X$ -pos	Part-of-speech annotations for $X$
$X$ -gov	Governor of $X$ in the dependency path
$X$ -mod	Modifiers of $X$ in the dependency path
$X$ -has_age	$X$ is a NE, with an age attribute
$X$ -has_class- $C$	$X$ is a NE, with a class $C$
$X$ -property- $P$	$X$ is a NE, and it has a property $P$
$X$ -has- $Y$	$X$ is a NE, with a possessive relation with another NE, $Y$
$X$ -is- $Y$	$X$ is a NE, in a copula with another NE, $Y$
$X$ -gender- $G$	$X$ is a NE, and it has gender $G$
$V$ -tense	Tense of the verb $V$ in the path
$V$ -aspect	Aspect of the verb $V$ in the path
$V$ -polarity	Polarity (positive or negative) of the verb $V$

Table 1: Features included in the model.  $X$  stands for ENTITY and VALUE. Verb features are generated from the verbs,  $V$ , identified in the path between ENTITY and VALUE.

we locate matching entities and start a local BFS of candidate values, generating for them an unlabelled example. For each of the relations to extract, a binary classifier (extractor) decides whether the example is a valid relation instance. For each particular relation classifier, only candidates with the expected entity and value types for the relation were used in the application phase. Each extractor was a SVM classifier with linear kernel (Joachims, 2002). All learning parameters were set to their default values.

The classification process yields a predicted class label, plus a real number indicating the margin. We performed an aggregation phase to sum the margins over distinct occurrences of the same extracted value. The rationale is that when the same value is extracted from more than one document, we should accumulate that evidence.

The output of this phase is the set of extracted relations (positive for each of the classifiers), plus the documents where the same fact was detected (*supporting documents*).

## 5 Temporal Anchoring of Relations

In this section, we propose and discuss a unified methodological approach for temporal anchoring of relations. We assume the input is a relation instance and a set of *supporting documents*. The task is establishing a imprecise temporal anchor interval for the relation.

We present a four-step methodological approach: (1) representation of intra-document temporal information; (2) selection of relevant temporal information for the relation; (3) mapping of the link between relational fact and temporal information into an interval; and (4) aggregation of imprecise intervals.

**Temporal representation.** The first methodological step is to obtain and represent the available intra-document temporal information; the input is a document, and the task is to identify temporal signals and possible *links* among them. We use the term *link* for a relation between a temporal expression (a date) and an event; we want to avoid confusion with the term *relation* (a relational fact extracted from text).

In our particular implementation:

- We use TARSQI to extract temporal expressions and link them to events. In particular, TARSQI uses the following temporal links: *included*, *simultaneous*, *after*, *before*, *begun\_by* or *ended*.
- We focus also on the syntactic pattern [*Event-preposition-Time*] within the lexical context of the candidate entity and value.
- Both are normalized into one from a set of predefined temporal links: *within*, *throughout*, *beginning*, *ending*, *after* and *before*.

**Selection of temporal evidence.** For each document and relational instance, we have to select those temporal expressions that are relevant.

- Document-level metadata.** The default value we use is the *document creation time* (DCT), if available. The underlying assumption is that there is a *within* link from each fact expressed in the text and the document creation time.
- Temporal expressions.** Temporal evidence comes also from the temporal expressions present in the context of a relation. In our particular implementation, we followed a straightforward approach, looking for the time expression closest in the document graph to the shortest path between the entity and value nodes. This search is performed via a limited depth BFS, starting from the nodes in the path, in order from value to entity.

**Mapping of temporal links into intervals.** The third step is deciding how a relational fact and its relevant temporal information are themselves related. We have to map this information, expressed in text,

Temporal link	Constraints mapping
Before	$t_4 = first$
After	$t_1 = last$
Within and Throughout	$t_2 = first$ and $t_3 = last$
Beginning	$t_1 = first$ and $t_2 = last$
Ending	$t_3 = first$ and $t_4 = last$

Table 2: Mapping from time expression and temporal relation to temporal constraints.

to a temporal representation. We will use the imprecise anchor intervals described in section 2.

Let  $T$  be a temporal expression identified in the document or its metadata. Now, the mapping of temporal constraints depends on the temporal link to the time expression identified; also, the semantics of the event have to be considered in order to decide the *time period* associated to a relation instance. This step is important because the event could refer just to the beginning of the relation, its ending, or both. For instance, it is obvious that having the event *marry* is different to having the event *divorce*, when deciding the temporal constraints associated to the *spouse* relation.

Table 2 shows our particular mapping between temporal links and constraints. In particular, for the default document creation time, we suppose that a relation which appears in a document with creation time  $d$  held true at least in that date; that is, we are assuming a *within* link, and we map  $t_2 = d$ ,  $t_3 = d$ .

### Inter-document temporal evidence aggregation.

The last step is aggregating all the time constraints found for the same relation and value across different documents. If we found that a relation started after two dates  $d$  and  $d'$ , where  $d' > d$ , the closest constraint to the real start of the relation is  $d'$ . Mapped to temporal constraints, it means that we would choose the biggest  $t_1$  possible. Following the same reasoning, we would want to maximize  $t_3$ . On the other side, when a relation started before two dates  $d_2$  and  $d'_2$ , where  $d'_2 > d_2$ , the closest constraint is  $d_2$  and we would choose the smallest  $t_2$ . In summary, we will maximize  $t_1$  and  $t_3$  and minimize  $t_2$  and  $t_4$ , so we will narrow the margins.

## 6 Evaluation

We have used for our evaluation the dataset compiled within the TAC-KBP 2011 Temporal Slot Filling Task (Ji et al., 2011). We employed as initial

KB the one distributed to participants in the task, which has been compiled from Wikipedia infoboxes. It contains 898 triples  $\langle entity, slot\_type, value \rangle$  for 100 different entities and up to 8 different slots (relations) per entity<sup>3</sup>. This gold standard contains the correct responses pooled from the participant systems plus a set of responses manually found by annotators. Each triple has associated a temporal anchor. The relations had to be extracted from a domain-general collection of 1.7 million documents. Our system was one of the five that took part in the task. We have evaluated the overall system and the two main components of the architecture: Relation Extraction, and Temporal Anchoring of the relations. Due to space limitations, the description of our implementation is very concise; refer to (Garrido et al., 2011) for further details.

### 6.1 Evaluation of Relation Extraction

System response in the relation extraction step consists in a set of triples  $\langle entity, slot\_type, value \rangle$ . Performance is measured using precision, recall and F-measure (harmonic mean) with respect to the 898 triples in the key. Target relations (slots) are potentially *list-valued*, that is, more than one value can be valid for a relation (possibly at different points in time). Only correct values yield any score, and redundant triples are ignored.

**Experiments.** We run two different system settings for the relation extraction step. They differ in the document representation used (detailed in section 3), in order to empirically assess whether clustering of discourse referents into single nodes benefits the extraction. In SETTING 1, each document is represented as a document graph,  $G_D$ , while in SETTING 2 collapsed document graph representation,  $G_C$ , is employed.

**Results.** Results are shown in Table 3 in the column *Relation Extraction*. Both settings have a similar performance with a slight increase in the case of graphs with clustered referents. Although precision is close to 0.5, recall is lower than 0.1. We have studied the limits of the assumptions our approach

<sup>3</sup>There are 7 *person* relations: *cities\_of\_residence*, *state-or-provinces\_of\_residence*, *countries\_of\_residence*, *employee\_of*, *member\_of*, *title*, *spouse*, and an *organization* relation: *top\_members/employees*.

is based on. First, our standard retrieval component performance limits the overall system’s. As a matter of example, if we retrieve the first 100 documents per entity, we find relevant documents only for 62% of the triples in the key. This number means that no matter how good relation extraction method is, 38% of relations will not be found.

Second, the *distant supervision assumption* underlying our approach is that for a seed relation instance  $\langle \text{entity}, \text{relation}, \text{value} \rangle$ , any textual mention of *entity* and *value* expresses the *relation*. It has been shown that this assumption is more often violated when training knowledge base and document collection are of different type, e.g. Wikipedia and news-wire (Riedel et al., 2010). We have realized that a more determinant factor is the relation itself and the type of arguments it takes. We randomly sampled 100 training examples per relation, and manually inspected them to assess if they were indeed mentions of the relation. While for the relation *cities\_of\_residence* only 30% of the training examples are expressing the relation, for *spouse* the number goes up to 59%. For *title*, up to 90% of the examples are correct. This fact explains, at least partially, the zeros we obtain for some relations.

## 6.2 Evaluation of Temporal Anchoring

Under the evaluation metrics proposed by TAC-KBP 2011, if the value of the relation instance is judged as correct, the score for temporal anchoring depends on how well the returned interval matches the one provided in the key. More precisely, let the correct imprecise anchor interval in the gold standard key be  $S_k = (k_1, k_2, k_3, k_4)$  and the system response be  $S = (r_1, r_2, r_3, r_4)$ . The absence of a constraint in  $t_1$  or  $t_3$  is treated as a value of  $-\infty$ ; the absence of a constraint in  $t_2$  or  $t_4$  is treated as a value of  $+\infty$ . Then, let  $d_i = |k_i - r_i|$ , for  $i \in 1, \dots, 4$ , be the difference, a real number *measured in years*. The score for the system response is:

$$Q(S) = \frac{1}{4} \sum_{i=1}^4 \frac{1}{1 + d_i}$$

The score for a target relation  $Q(r)$  is computed by summing  $Q(S)$  over all unique instances of the relation whose value is correct. If the gold standard contains  $N$  responses, and the system output  $M$  responses, then precision is:  $P = Q(r)/M$ , and recall:

$R = Q(r)/N$ ;  $F_1$  is the harmonic mean of  $P$  and  $R$ .

**Experiments.** We evaluated two different settings for the temporal anchoring step; both use the collapsed document graph representation,  $G_C$  (SETTING 2). The goal of the experiment is two-fold. First, test the strength of the *document creation time* as evidence for temporal anchoring. Second, test how hard this metadata-level baseline is to beat using contextual temporal expressions.

The SETTING 2-I assumes a *within* temporal link between the document creation time and any relation expressed inside the document, and aggregates this information across the documents that we have identified as supporting the relation. The SETTING 2-II considers documents content in order to extract temporal links from the context of the text that expresses the relation. If no temporal expression is found, the date of the document is used as default. Temporal links from all supporting documents are mapped into intervals and aggregated as detailed in section 5.

The performance on relation extraction is an upper bound for temporal anchoring, attainable if temporal anchoring is perfect. Thus, we also evaluate the temporal anchoring performance as the percentage the final system achieves with respect to the relation extraction upper bound.

**Results.** Results are shown in Table 3 under column *Temporal Anchoring*. They are low, due to the upper bound that error propagation in candidate retrieval and relation extraction imposes upon this step: temporally anchoring alone achieves 69% of its upper bound. This value corresponds to the baseline SETTING 2-I, showing its strength. The difference with SETTING 2-II shows that this baseline is difficult to beat by considering temporal evidence inside the document content. There is a reason for this. The temporal link mapping into time intervals does not depend only on the type of link, but also on the semantics of the text that expresses the relation as we pointed out above. We have to decide how to transform the link between relation and temporal expression into a temporal interval. Learning a model for this is a hard open research problem that has a strong adversary in the baseline proposed.

	Relation Extraction						Temporal Anchoring							
	SETTING 1			SETTING 2			SETTING 2-I				SETTING 2-II			
	P	R	F	P	R	F	P	R	F	%	P	R	F	%
(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0
(3)	0.33	0.02	0.03	0	0	0	0	0	0	0	0	0	0	0
(4)	0.22	0.09	0.13	0.29	0.11	0.16	0.23	0.09	0.13	79	0.21	0.08	0.11	72
(5)	0.53	0.13	0.20	0.54	0.12	0.19	0.34	0.07	0.12	63	0.30	0.06	0.11	56
(6)	0.70	0.12	0.20	0.75	0.13	0.22	0.57	0.10	0.16	76	0.50	0.08	0.14	67
(7)	0.50	0.06	0.10	0.50	0.07	0.12	0.29	0.04	0.07	58	0.25	0.04	0.06	50
(8)	0.25	0.04	0.07	0.20	0.04	0.07	0.15	0.03	0.05	75	0.06	0.01	0.02	30
(9)	0.42	0.08	0.14	0.45	0.08	0.14	0.31	0.06	0.10	69	0.27	0.05	0.09	60

Table 3: Results of experiments for each relation: (1) per:stateorprovinces\_of\_residence; (2) per:employee\_of; (3) per:countries\_of\_residence; (4) per:member\_of; (5) per:title; (6) org:top\_members/employees; (7) per:spouse; (8) per:cities\_of\_residence; (9) overall results (calculated as a micro-average).

System	# Filled	Precision	Recall	F1
BLENDER2	1206	0.1789	0.3030	0.2250
BLENDER1	1116	0.1796	0.2942	0.2231
BLENDER3	1215	0.1744	0.2976	0.2199
IIRG1	346	0.2457	0.1194	0.1607
<b>Setting 2-1</b>	<b>167</b>	<b>0.2996</b>	<b>0.0703</b>	<b>0.1139</b>
<b>Setting 2-2</b>	<b>167</b>	<b>0.2596</b>	<b>0.0609</b>	<b>0.0986</b>
Stanford 12	5140	0.0233	0.1680	0.0409
Stanford 11	4353	0.0238	0.1453	0.0408
USFD20112	328	0.0152	0.0070	0.0096
USFD20113	127	0.0079	0.0014	0.0024

Table 4: System ID, number of filled responses of the system, precision, recall and F measure.

### 6.3 Comparative Evaluation

Our approach was compared with the other four participants at the KBP Temporal Slot Filling Task 2011. Table 4 shows results sorted by F-measure in comparison to our two settings (described above). These official results correspond to a previous dataset containing 712 triples<sup>4</sup>.

As shown in column *Filled* our approach returns less triples than other systems, explaining low recall. However, our system achieves the highest precision for the complete task of temporally anchored relation extraction. Despite low recall, our system obtains the third best  $F_1$  value. This is a very promising result, since several directions can be explored to consider more candidates and increase recall.

## 7 Related Work

Compiling a Knowledge Base of temporally anchored facts is an open research challenge (Weikum et al., 2011). Despite the vast amount of research focusing on understanding temporal expressions and

their relation to events in natural language, the complete problem of temporally anchored relation extraction remains relatively unexplored. Also, while much research has focused on single-document extraction, it seems clear that extracting temporally anchored relations needs the aggregation of evidences across multiple documents.

There have been attempts to extend an existing knowledge base. Wang et al. (2010) use regular expressions to mine Wikipedia infoboxes and categories and it is not suited for unrestricted text. An earlier attempt (Zhang et al., 2008), is specific for business and difficult to generalize to other relations. Two recent promising works are more related to our research. Wang et al. (2011) uses manually defined patterns to collect candidate facts and explicit dates, and re-rank them using a graph label propagation algorithm; their approach is complementary to ours, as our aim is not to harvest temporal facts but to extract the relations in which a query entity takes part; unlike us, they require entity, value, and an explicit date to appear in the same sentence. Talukdar et al. (2012) focus on the partial task of temporally anchoring already known facts, showing the usefulness of the document creation time as temporal signal, aggregated across documents.

Earlier work has dealt mainly with partial aspects of the problem. The TempEval community focused on the classification of the temporal links between pairs of events, or an event and a temporal expression; using shallow features (Mani et al., 2003; Lapata and Lascarides, 2004; Chambers et al., 2007), or syntactic-based structured features (Bethard and Martin, 2007; Puşcaşu, 2007; Cheng et al., 2007).

Aggregating evidence across different documents

<sup>4</sup>Slot-fillers from human assessors were not considered

to temporally anchor facts has been explored in settings different to Information Extraction, such as answering of definition questions (Paşca, 2008) or extracting possible dates of well-known historical events (Schockaert et al., 2010).

Temporal inference or reasoning to solve conflicting temporal expressions and induce temporal order of events has been used in TempEval (Tatu and Srikanth, 2008; Yoshikawa et al., 2009) and ACE (Gupta and Ji, 2009) tasks, but focused on single-document extraction. Ling et al. (2010), use cross-event joint inference to extract temporal facts, but only inside a single document.

Evaluation campaigns, such as ACE and TAC-KBP 2011 have had an important role in promoting this research. While ACE required only to identify time expressions and classify their relation to events, KBP requires to infer explicitly the start/end time of relations, which is a realistic approach in the context of building time-aware knowledge bases. KBP represents an important step for the evaluation of temporal information extraction systems. In general, the participant systems adapted existing slot filling systems, adding a temporal classification component: distant supervised (Chen et al., 2010; Surdeanu et al., 2010) on manually-defined patterns (Byrne and Dunnion, 2010).

## 8 Conclusions

This paper introduces the problem of extracting, from unrestricted natural language text, relational knowledge anchored to a temporal span, aggregating temporal evidence from a collection of documents. Although compiling time-aware knowledge bases is an important open challenge (Weikum et al., 2011), it has remained unexplored until very recently (Wang et al., 2011; Talukdar et al., 2012).

We have elucidated the two challenges of the task, namely relation extraction and temporal anchoring of the extracted relations.

We have studied how, in a pipeline architecture, the propagation of errors limits the overall system’s performance. The performance attainable in the full task is limited by the quality of the output of the three main phases: retrieval of candidate passages/documents, extraction of relations and temporal anchoring of those.

We have also studied the limits of the distant supervision approach to relation extraction, showing empirically that its performance depends not only on the nature of reference knowledge base and document corpus (Riedel et al., 2010), but also on the relation to be extracted. Given a relation between two arguments, if it is not dominant among textual expressions of those arguments, the distant supervision assumption will be more often violated.

We have introduced a novel graph-based document level representation, that has allowed us to generate new features for the task of relation extraction, capturing long distance structured contexts. Our results show how, in a document level syntactic representation, it yields better results to collapse coreferent nodes.

We have presented a methodological approach to temporal anchoring composed of: (1) intra-document temporal information representation; (2) selection of relation-dependent relevant temporal information; (3) mapping of temporal links to an interval representation; and (4) aggregation of imprecise intervals.

Our proposal has been evaluated within a framework that allows for comparability. It has been able to extract temporally anchored relational information with the highest precision among the participant systems taking part in the competitive evaluation TAC-KBP 2011.

For the temporal anchoring sub-problem, we have demonstrated the strength of the document creation time as a temporal signal. It is possible to achieve a performance of 69% of the upper-bound imposed by relation extraction by assuming that any relation mentioned in a document held at the document creation time (there is a *within* link between the relational fact and the document creation time). This baseline has proved stronger than extracting and analyzing the temporal expressions present in the document content.

## Acknowledgments

This work has been partially supported by the Spanish Ministry of Science and Innovation, through the project Holopedia (TIN2010-21128-C02), and the Regional Government of Madrid, through the project MA2VICMR (S2009/TIC1542).

## References

- Eneko Agirre, Angel X. Chang, Daniel S. Jurafsky, Christopher D. Manning, Valentin I. Spitzkovsky, and Eric Yeh. 2009. Stanford-UBC at TAC-KBP. In *TAC 2009*, November.
- James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26:832–843, November.
- Steven Bethard and James H. Martin. 2007. Cu-tmp: temporal relation classification using syntactic and semantic features. In *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval '07*, pages 129–132, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lorna Byrne and John Dunnion. 2010. UCD IIRG at TAC 2010 KBP Slot Filling Task. In *Proceedings of the Third Text Analysis Conference (TAC 2010)*. NIST, November.
- Nathanael Chambers, Shan Wang, and Dan Jurafsky. 2007. Classifying temporal relations between events. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, ACL '07*, pages 173–176, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew Snover, Javier Artiles, Marissa Passantino, and Heng Ji. 2010. CUNY-BLENDER TAC-KBP2010: Entity linking and slot filling system description. In *Proceedings of the Third Text Analysis Conference (TAC 2010)*. NIST, November.
- Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2007. Naist.japan: temporal relation identification using dependency parsed tree. In *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval '07*, pages 245–248, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Guillermo Garrido, Bernardo Cabaleiro, Anselmo Peas, varo Rodrigo, and Damiano Spina. 2011. A distant supervised learning system for the TAC-KBP Slot Filling and Temporal Slot Filling Tasks. In *Text Analysis Conference, TAC 2011 Proceedings Papers*.
- Prashant Gupta and Heng Ji. 2009. Predicting unknown time arguments based on cross-event propagation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, ACLShort '09*, pages 369–372, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Heng Ji, Ralph Grishman, and Hoa Trang Dang. 2011. Overview of the tac2011 knowledge base population track. In *Text Analysis Conference, TAC 2011 Workshop, Notebook Papers*.
- T. Joachims. 2002. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms*. Kluwer/Springer. We used Joachim's SVMlight implementation available at <http://svmlight.joachims.org/>.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL 2003*, pages 423–430.
- Mirella Lapata and Alex Lascarides. 2004. Inferring sentence-internal temporal relations. In *HLT 2004*.
- Xiao Ling and Daniel S. Weld. 2010. Temporal information extraction. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*.
- Inderjeet Mani, Barry Schiffman, and Jianping Zhang. 2003. Inferring temporal ordering of events in news. In *NAACL-Short'03*.
- Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL 2009*, pages 1003–1011, Stroudsburg, PA, USA. Association for Computational Linguistics.
- M Paşca. 2008. Answering Definition Questions via Temporally-Anchored Text Snippets. *Proc. of IJCNLP2008*.
- Georgiana Puşcaşu. 2007. Wvali: temporal relation identification by syntactico-semantic analysis. In *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval '07*, pages 484–487, Stroudsburg, PA, USA. Association for Computational Linguistics.
- James Pustejovsky and Marc Verhagen. 2009. SemEval-2010 task 13: evaluating events, time expressions, and temporal relations (TempEval-2). In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions, DEW '09*, pages 112–116, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In José Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 6323 of *LNCS*, pages 148–163. Springer Berlin / Heidelberg.
- Stuart J. Russell and Peter Norvig. 2010. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education.
- Steven Schockaert, Martine De Cock, and Etienne Kerre. 2010. Reasoning about fuzzy temporal information from the web: towards retrieval of historical events. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, 14:869–886.
- Mihai Surdeanu and Massimiliano Ciaramita. 2007. Robust information extraction with perceptrons. In *ACE07*, March.

- Mihai Surdeanu, David McClosky, Julie Tibshirani, John Bauer, Angel X. Chang, Valentin I. Spitzkovsky, and Christopher D. Manning. 2010. A simple distant supervision approach for the tac-kbp slot filling task. In *Proceedings of the Third Text Analysis Conference (TAC 2010)*, Gaithersburg, Maryland, USA, November. NIST.
- Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. 2012. Coupled temporal scoping of relational facts. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM)*, Seattle, Washington, USA, February. Association for Computing Machinery.
- Marta Tatu and Munirathnam Srikanth. 2008. Experiments with reasoning for temporal relations between events. In *COLING'08*.
- Marc Verhagen, Inderjeet Mani, Roser Sauri, Robert Knippen, Seok Bae Jang, Jessica Littman, Anna Rumshisky, John Phillips, and James Pustejovsky. 2005. Automating temporal annotation with TARSQI. In *ACLdemo'05*.
- Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. SemEval-2007 task 15: TempEval temporal relation identification. In *SemEval'07*.
- Yafang Wang, Mingjie Zhu, Lizhen Qu, Marc Spaniol, and Gerhard Weikum. 2010. Timely YAGO: harvesting, querying, and visualizing temporal knowledge from Wikipedia. In *Proceedings of the 13th International Conference on Extending Database Technology, EDBT '10*, pages 697–700, New York, NY, USA. ACM.
- Yafang Wang, Bin Yang, Lizhen Qu, Marc Spaniol, and Gerhard Weikum. 2011. Harvesting facts from textual web sources by constrained label propagation. In *Proceedings of the 20th ACM international conference on Information and knowledge management, CIKM '11*, pages 837–846, New York, NY, USA. ACM.
- Gerhard Weikum, Srikanta Bedathur, and Ralf Schenkel. 2011. Temporal knowledge for timely intelligence. In Malu Castellanos, Umeshwar Dayal, Volker Markl, Wil Aalst, John Mylopoulos, Michael Rosemann, Michael J. Shaw, and Clemens Szyperski, editors, *Enabling Real-Time Business Intelligence*, volume 84 of *Lecture Notes in Business Information Processing*, pages 1–6. Springer Berlin Heidelberg.
- Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with Markov Logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1, ACL '09*, pages 405–413, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Qi Zhang, Fabian M. Suchanek, Lihua Yue, and Gerhard Weikum. 2008. TOB: Timely ontologies for business relations. In *11th International Workshop on the Web and Databases, WebDB*.



# Efficient Tree-based Approximation for Entailment Graph Learning

Jonathan Berant<sup>§</sup>, Ido Dagan<sup>†</sup>, Meni Adler<sup>‡</sup>, Jacob Goldberger<sup>‡</sup>

<sup>§</sup> The Blavatnik School of Computer Science, Tel Aviv University

<sup>†</sup> Department of Computer Science, Bar-Ilan University

<sup>‡</sup> Faculty of Engineering, Bar-Ilan University

jonatha6@post.tau.ac.il

{dagan, goldbej}@{cs, eng}.biu.ac.il

adlerm@cs.bgu.ac.il

## Abstract

Learning entailment rules is fundamental in many semantic-inference applications and has been an active field of research in recent years. In this paper we address the problem of learning transitive graphs that describe entailment rules between predicates (termed *entailment graphs*). We first identify that entailment graphs exhibit a “tree-like” property and are very similar to a novel type of graph termed *forest-reducible graph*. We utilize this property to develop an iterative efficient approximation algorithm for learning the graph edges, where each iteration takes linear time. We compare our approximation algorithm to a recently-proposed state-of-the-art exact algorithm and show that it is more efficient and scalable both theoretically and empirically, while its output quality is close to that given by the optimal solution of the exact algorithm.

## 1 Introduction

Performing textual inference is in the heart of many semantic inference applications such as Question Answering (QA) and Information Extraction (IE). A prominent generic paradigm for textual inference is Textual Entailment (TUE) (Dagan et al., 2009). In TUE, the goal is to recognize, given two text fragments termed *text* and *hypothesis*, whether the hypothesis can be inferred from the text. For example, the text “Cyprus was invaded by the Ottoman Empire in 1571” implies the hypothesis “The Ottomans attacked Cyprus”.

Semantic inference applications such as QA and IE crucially rely on *entailment rules* (Ravichandran

and Hovy, 2002; Shinyama and Sekine, 2006) or equivalently *inference rules*, that is, rules that describe a directional inference relation between two fragments of text. An important type of entailment rule specifies the entailment relation between natural language *predicates*, e.g., the entailment rule ‘ $X \text{ invade } Y \rightarrow X \text{ attack } Y$ ’ can be helpful in inferring the aforementioned hypothesis. Consequently, substantial effort has been made to learn such rules (Lin and Pantel, 2001; Sekine, 2005; Szpektor and Dagan, 2008; Schoenmackers et al., 2010).

Textual entailment is inherently a *transitive* relation, that is, the rules ‘ $x \rightarrow y$ ’ and ‘ $y \rightarrow z$ ’ imply the rule ‘ $x \rightarrow z$ ’. Accordingly, Berant et al. (2010) formulated the problem of learning entailment rules as a graph optimization problem, where nodes are predicates and edges represent entailment rules that respect transitivity. Since finding the optimal set of edges respecting transitivity is NP-hard, they employed Integer Linear Programming (ILP) to find the exact solution. Indeed, they showed that applying global transitivity constraints improves rule learning comparing to methods that ignore graph structure. More recently, Berant et al. (Berant et al., 2011) introduced a more efficient exact algorithm, which decomposes the graph into connected components and then applies an ILP solver over each component.

Despite this progress, finding the exact solution remains NP-hard – the authors themselves report they were unable to solve some graphs of rather moderate size and that the coverage of their method is limited. Thus, scaling their algorithm to data sets with tens of thousands of predicates (e.g., the extractions of Fader et al. (2011)) is unlikely.

In this paper we present a novel method for learning the edges of entailment graphs. Our method computes much more efficiently an approximate solution that is empirically almost as good as the exact solution. To that end, we first (Section 3) conjecture and empirically show that entailment graphs exhibit a “tree-like” property, i.e., that they can be *reduced* into a structure similar to a directed forest.

Then, we present in Section 4 our iterative approximation algorithm, where in each iteration a node is removed and re-attached back to the graph in a locally-optimal way. Combining this scheme with our conjecture about the graph structure enables a linear algorithm for node re-attachment. Section 5 shows empirically that this algorithm is by orders of magnitude faster than the state-of-the-art exact algorithm, and that though an optimal solution is not guaranteed, the area under the precision-recall curve drops by merely a point.

To conclude, the contribution of this paper is twofold: First, we define a novel modeling assumption about the tree-like structure of entailment graphs and demonstrate its validity. Second, we exploit this assumption to develop a polynomial approximation algorithm for learning entailment graphs that can scale to much larger graphs than in the past. Finally, we note that learning entailment graphs bears strong similarities to related tasks such as Taxonomy Induction (Snow et al., 2006) and Ontology induction (Poon and Domingos, 2010), and thus our approach may improve scalability in these fields as well.

## 2 Background

Until recently, work on learning entailment rules between predicates considered each rule independently of others and did not exploit global dependencies. Most methods utilized the distributional similarity hypothesis that states that semantically similar predicates occur with similar arguments (Lin and Pantel, 2001; Szpektor et al., 2004; Yates and Etzioni, 2009; Schoenmackers et al., 2010). Some methods extracted rules from lexicographic resources such as WordNet (Szpektor and Dagan, 2009) or FrameNet (Bob and Rambow, 2009; Ben Aharon et al., 2010), and others assumed that semantic relations between predicates can be deduced from their co-occurrence in a corpus via manually-constructed

patterns (Chklovski and Pantel, 2004).

Recently, Berant et al. (2010; 2011) formulated the problem as the problem of learning global *entailment graphs*. In entailment graphs, nodes are predicates (e.g., ‘*X attack Y*’) and edges represent entailment rules between them (‘*X invade Y*  $\rightarrow$  *X attack Y*’). For every pair of predicates  $i, j$ , an entailment score  $w_{ij}$  was learned by training a classifier over distributional similarity features. A positive  $w_{ij}$  indicated that the classifier believes  $i \rightarrow j$  and a negative  $w_{ij}$  indicated that the classifier believes  $i \not\rightarrow j$ . Given the graph nodes  $V$  (corresponding to the predicates) and the weighting function  $w : V \times V \rightarrow \mathbb{R}$ , they aim to find the edges of a graph  $\mathcal{G} = (V, E)$  that maximize the objective  $\sum_{(i,j) \in E} w_{ij}$  under the constraint that the graph is *transitive* (i.e., for every node triplet  $(i, j, k)$ , if  $(i, j) \in E$  and  $(j, k) \in E$ , then  $(i, k) \in E$ ).

Berant et al. proved that this optimization problem, which we term *Max-Trans-Graph*, is NP-hard, and so described it as an Integer Linear Program (ILP). Let  $x_{ij}$  be a binary variable indicating the existence of an edge  $i \rightarrow j$  in  $E$ . Then,  $\mathcal{X} = \{x_{ij} : i \neq j\}$  are the variables of the following ILP for Max-Trans-Graph:

$$\begin{aligned} \arg \max_{\mathcal{X}} \quad & \sum_{i \neq j} w_{ij} \cdot x_{ij} & (1) \\ \text{s.t.} \quad & \forall_{i,j,k \in V} x_{ij} + x_{jk} - x_{ik} \leq 1 \\ & \forall_{i,j \in V} x_{ij} \in \{0, 1\} \end{aligned}$$

The objective function is the sum of weights over the edges of  $\mathcal{G}$  and the constraint  $x_{ij} + x_{jk} - x_{ik} \leq 1$  on the binary variables enforces that whenever  $x_{ij} = x_{jk} = 1$ , then also  $x_{ik} = 1$  (transitivity).

Since ILP is NP-hard, applying an ILP solver directly does not scale well because the number of variables is  $O(|V|^2)$  and the number of constraints is  $O(|V|^3)$ . Thus, even a graph with  $\sim 80$  nodes (predicates) has more than half a million constraints. Consequently, in (Berant et al., 2011), they proposed a method that efficiently decomposes the graph into smaller components and applies an ILP solver on each component separately using a cutting-plane procedure (Riedel and Clarke, 2006). Although this method is exact and improves scalability, it does not guarantee an efficient solution. When the graph does not decompose into sufficiently small components, and the weights generate many violations of

transitivity, solving Max-Trans-Graph becomes intractable. To address this problem, we present in this paper a method for approximating the optimal set of edges within each component and show that it is much more efficient and scalable both theoretically and empirically.

Do and Roth (2010) suggested a method for a related task of learning taxonomic relations between terms. Given a pair of terms, a small graph is constructed and constraints are imposed on the graph structure. Their work, however, is geared towards scenarios where relations are determined on-the-fly for a given pair of terms and no global knowledge base is explicitly constructed. Thus, their method easily produces solutions where global constraints, such as transitivity, are violated.

Another approximation method that violates transitivity constraints is LP relaxation (Martins et al., 2009). In LP relaxation, the constraint  $x_{ij} \in \{0, 1\}$  is replaced by  $0 \leq x_{ij} \leq 1$ , transforming the problem from an ILP to a Linear Program (LP), which is polynomial. An LP solver is then applied on the problem, and variables  $x_{ij}$  that are assigned a fractional value are rounded to their nearest integer and so many violations of transitivity easily occur. The solution when applying LP relaxation is not a transitive graph, but nevertheless we show for comparison in Section 5 that our method is much faster.

Last, we note that transitive relations have been explored in adjacent fields such as Temporal Information Extraction (Ling and Weld, 2010), Ontology Induction (Poon and Domingos, 2010), and Co-reference Resolution (Finkel and Manning, 2008).

### 3 Forest-reducible Graphs

The entailment relation, described by entailment graphs, is typically from a “semantically-specific” predicate to a more “general” one. Thus, intuitively, the topology of an entailment graph is expected to be “tree-like”. In this section we first formalize this intuition and then empirically analyze its validity. This property of entailment graphs is an interesting topological observation on its own, but also enables the efficient approximation algorithm of Section 4.

For a directed edge  $i \rightarrow j$  in a directed acyclic graphs (DAG), we term the node  $i$  a *child* of node  $j$ , and  $j$  a *parent* of  $i$ . A *directed forest* is a DAG

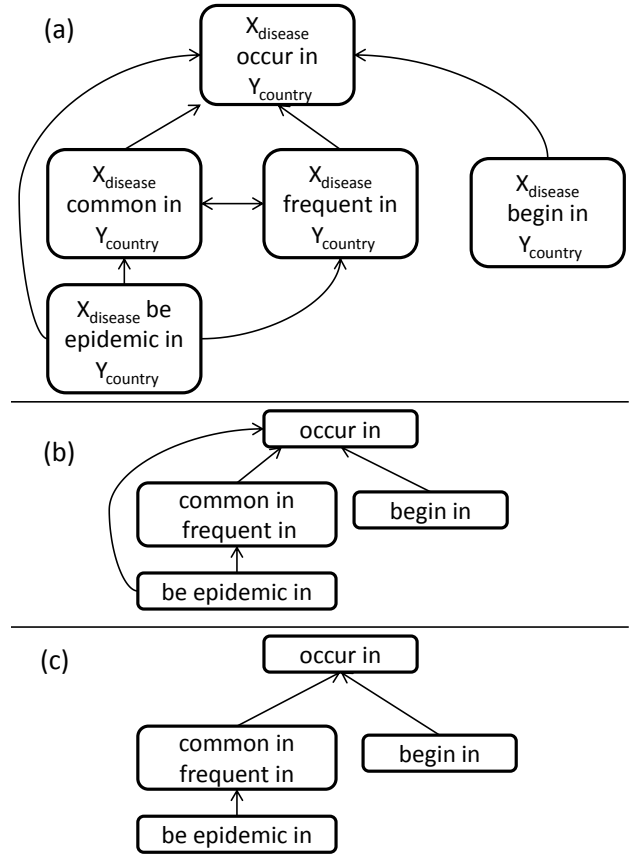


Figure 1: A fragment of an entailment graph (a), its SCC graph (b) and its reduced graph (c). Nodes are predicates with typed variables (see Section 5), which are omitted in (b) and (c) for compactness.

where all nodes have no more than one parent.

The entailment graph in Figure 1a (subgraph from the data set described in Section 5) is clearly not a directed forest – it contains a cycle of size two comprising the nodes ‘ $X \text{ common in } Y$ ’ and ‘ $X \text{ frequent in } Y$ ’, and in addition the node ‘ $X \text{ be epidemic in } Y$ ’ has 3 parents. However, we can convert it to a directed forest by applying the following operations. Any directed graph  $\mathcal{G}$  can be converted into a *Strongly-Connected-Component (SCC)* graph in the following way: every strongly connected component (a set of semantically-equivalent predicates, in our graphs) is contracted into a single node, and an edge is added from SCC  $S_1$  to SCC  $S_2$  if there is an edge in  $\mathcal{G}$  from some node in  $S_1$  to some node in  $S_2$ . The SCC graph is always a DAG (Cormen et al., 2002), and if  $\mathcal{G}$  is transitive then the SCC graph is also transitive. The graph in Figure 1b is the SCC graph of the one in

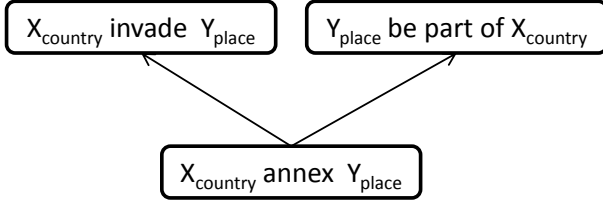


Figure 2: A fragment of an entailment graph that is not an FRG.

Figure 1a, but is still not a directed forest since the node ‘ $X$  be epidemic in  $Y$ ’ has two parents.

The *transitive closure* of a directed graph  $\mathcal{G}$  is obtained by adding an edge from node  $i$  to node  $j$  if there is a path in  $\mathcal{G}$  from  $i$  to  $j$ . The *transitive reduction* of  $\mathcal{G}$  is obtained by removing all edges whose absence does not affect its transitive closure. In DAGs, the result of transitive reduction is unique (Aho et al., 1972). We thus define the *reduced graph*  $\mathcal{G}_{red} = (V_{red}, E_{red})$  of a directed graph  $\mathcal{G}$  as the transitive reduction of its SCC graph. The graph in Figure 1c is the reduced graph of the one in Figure 1a and is a directed forest. We say a graph is a *forest-reducible graph (FRG)* if all nodes in its reduced form have no more than one parent.

We now hypothesize that entailment graphs are FRGs. The intuition behind this assumption is that the predicate on the left-hand-side of a unidirectional entailment rule has a more specific meaning than the one on the right-hand-side. For instance, in Figure 1a ‘ $X$  be epidemic in  $Y$ ’ (where ‘ $X$ ’ is a type of disease and ‘ $Y$ ’ is a country) is more specific than ‘ $X$  common in  $Y$ ’ and ‘ $X$  frequent in  $Y$ ’, which are equivalent, while ‘ $X$  occur in  $Y$ ’ is even more general. Accordingly, the reduced graph in Figure 1c is an FRG. We note that this is not always the case: for example, the entailment graph in Figure 2 is not an FRG, because ‘ $X$  annex  $Y$ ’ entails both ‘ $Y$  be part of  $X$ ’ and ‘ $X$  invade  $Y$ ’, while the latter two do not entail one another. However, we hypothesize that this scenario is rather uncommon. Consequently, a natural variant of the Max-Trans-Graph problem is to restrict the required output graph of the optimization problem (1) to an FRG. We term this problem *Max-Trans-Forest*.

To test whether our hypothesis holds empirically we performed the following analysis. We sampled 7 gold standard entailment graphs from the data set

described in Section 5, manually transformed them into FRGs by deleting a minimal number of edges, and measured recall over the set of edges in each graph (precision is naturally 1.0, as we only delete gold standard edges). The lowest recall value obtained was 0.95, illustrating that deleting a very small proportion of edges converts an entailment graph into an FRG. Further support for the practical validity of this hypothesis is obtained from our experiments in Section 5. In these experiments we show that exactly solving Max-Trans-Graph and Max-Trans-Forest (with an ILP solver) results in nearly identical performance.

An ILP formulation for Max-Trans-Forest is simple – a transitive graph is an FRG if all nodes in its reduced graph have no more than one parent. It can be verified that this is equivalent to the following statement: for every triplet of nodes  $i, j, k$ , if  $i \rightarrow j$  and  $i \rightarrow k$ , then either  $j \rightarrow k$  or  $k \rightarrow j$  (or both). Therefore, the ILP is formulated by adding this linear constraint to ILP (1):

$$\forall_{i,j,k \in V} x_{ij} + x_{ik} + (1 - x_{jk}) + (1 - x_{kj}) \leq 3 \quad (2)$$

We note that despite the restriction to FRGs, Max-Trans-Forest is an NP-hard problem by a reduction from the X3C problem (Garey and Johnson, 1979). We omit the reduction details for brevity.

## 4 Sequential Approximation Algorithms

In this section we present *Tree-Node-Fix*, an efficient approximation algorithm for Max-Trans-Forest, as well as *Graph-Node-Fix*, an approximation for Max-Trans-Graph.

### 4.1 Tree-Node-Fix

The scheme of Tree-Node-Fix (TNF) is the following. First, an initial FRG is constructed, using some initialization procedure. Then, at each iteration a single node  $v$  is *re-attached* (see below) to the FRG in a way that improves the objective function. This is repeated until the value of the objective function cannot be improved anymore by re-attaching a node.

*Re-attaching* a node  $v$  is performed by removing  $v$  from the graph and connecting it back with a better set of edges, while maintaining the constraint that it is an FRG. This is done by considering all possible edges from/to the other graph nodes and choosing

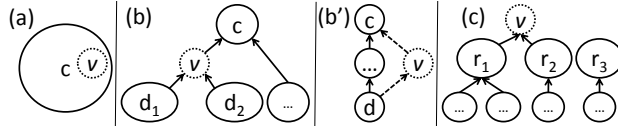


Figure 3: (a) Inserting  $v$  into a component  $c \in V_{red}$ . (b) Inserting  $v$  as a child of  $c$  and a parent of a subset of  $c$ 's children in  $\mathcal{G}_{red}$ . (b') A node  $d$  that is a descendant but not a child of  $c$  can not choose  $v$  as a parent, as  $v$  becomes its second parent. (c) Inserting  $v$  as a new root.

the *optimal* subset, while the rest of the graph remains fixed. Formally, let  $S_{v-in} = \sum_{i \neq v} w_{iv} \cdot x_{iv}$  be the sum of scores over  $v$ 's incoming edges and  $S_{v-out} = \sum_{k \neq v} w_{vk} \cdot x_{vk}$  be the sum of scores over  $v$ 's outgoing edges. Re-attachment amounts to optimizing a linear objective:

$$\arg \max_{\mathcal{X}_v} (S_{v-in} + S_{v-out}) \quad (3)$$

where the variables  $\mathcal{X}_v \subseteq \mathcal{X}$  are indicators for all pairs of nodes involving  $v$ . We approximate a solution for (1) by iteratively optimizing the simpler objective (3). Clearly, at each re-attachment the value of the objective function cannot decrease, since the optimization algorithm considers the previous graph as one of its candidate solutions.

We now show that re-attaching a node  $v$  is linear. To analyze  $v$ 's re-attachment, we consider the structure of the directed forest  $\mathcal{G}_{red}$  just *before*  $v$  is re-inserted, and examine the possibilities for  $v$ 's insertion relative to that structure. We start by defining some helpful notations. Every node  $c \in V_{red}$  is a connected component in  $\mathcal{G}$ . Let  $v_c \in c$  be an arbitrary representative node in  $c$ . We denote by  $S_{v-in}(c)$  the sum of weights from all nodes in  $c$  and their descendants to  $v$ , and by  $S_{v-out}(c)$  the sum of weights from  $v$  to all nodes in  $c$  and their ancestors:

$$S_{v-in}(c) = \sum_{i \in c} w_{iv} + \sum_{k \notin c} w_{kv} x_{kv_c}$$

$$S_{v-out}(c) = \sum_{i \in c} w_{vi} + \sum_{k \notin c} w_{vk} x_{v_c k}$$

Note that  $\{x_{v_c k}, x_{kv_c}\}$  are edge indicators in  $\mathcal{G}$  and not  $\mathcal{G}_{red}$ . There are two possibilities for re-attaching  $v$  – either it is inserted into an existing component  $c \in V_{red}$  (Figure 3a), or it forms a new component. In the latter, there are also two cases: either  $v$  is inserted as a child of a component  $c$  (Fig-

ure 3b), or not and then it becomes a root in  $\mathcal{G}_{red}$  (Figure 3c). We describe the details of these 3 cases:

**Case 1:** Inserting  $v$  into a component  $c \in V_{red}$ . In this case we add in  $\mathcal{G}$  edges from all nodes in  $c$  and their descendants to  $v$  and from  $v$  to all nodes in  $c$  and their ancestors. The score (3) in this case is

$$s_1(c) \triangleq S_{v-in}(c) + S_{v-out}(c) \quad (4)$$

**Case 2:** Inserting  $v$  as a child of some  $c \in V_{red}$ . Once  $c$  is chosen as the parent of  $v$ , choosing  $v$ 's children in  $\mathcal{G}_{red}$  is substantially constrained. A node that is not a descendant of  $c$  can not become a child of  $v$ , since this would create a new path from that node to  $c$  and would require by transitivity to add a corresponding directed edge to  $c$  (but all graph edges not connecting  $v$  are fixed). Moreover, only a direct child of  $c$  can choose  $v$  as a parent instead of  $c$  (Figure 3b), since for any other descendant of  $c$ ,  $v$  would become a second parent, and  $\mathcal{G}_{red}$  will no longer be a directed forest (Figure 3b'). Thus, this case requires adding in  $\mathcal{G}$  edges from  $v$  to all nodes in  $c$  and their ancestors, and also for each new child of  $v$ , denoted by  $d \in V_{red}$ , we add edges from all nodes in  $d$  and their descendants to  $v$ . Crucially, although the number of possible subsets of  $c$ 's children in  $\mathcal{G}_{red}$  is exponential, the fact that they are independent trees in  $\mathcal{G}_{red}$  allows us to go over them one by one, and decide for each one whether it will be a child of  $v$  or not, depending on whether  $S_{v-in}(d)$  is positive. Therefore, the score (3) in this case is:

$$s_2(c) \triangleq S_{v-out}(c) + \sum_{d \in \text{child}(c)} \max(0, S_{v-in}(d)) \quad (5)$$

where  $\text{child}(c)$  are the children of  $c$ .

**Case 3:** Inserting  $v$  as a new root in  $\mathcal{G}_{red}$ . Similar to case 2, only roots of  $\mathcal{G}_{red}$  can become children of  $v$ . In this case for each chosen root  $r$  we add in  $\mathcal{G}$  edges from the nodes in  $r$  and their descendants to  $v$ . Again, each root can be examined independently. Therefore, the score (3) of re-attaching  $v$  is:

$$s_3 \triangleq \sum_r \max(0, S_{v-in}(r)) \quad (6)$$

where the summation is over the roots of  $\mathcal{G}_{red}$ .

It can be easily verified that  $S_{v-in}(c)$  and  $S_{v-out}(c)$  satisfy the recursive definitions:

---

**Algorithm 1** Computing optimal re-attachment

---

**Input:** FRG  $\mathcal{G} = (V, E)$ , function  $w$ , node  $v \in V$ **Output:** optimal re-attachment of  $v$ 

- 1: remove  $v$  and compute  $G_{red} = (V_{red}, E_{red})$ .
  - 2: for all  $c \in V_{red}$  in post-order compute  $S_{v-in}(c)$  (Eq. 7)
  - 3: for all  $c \in V_{red}$  in pre-order compute  $S_{v-out}(c)$  (Eq. 8)
  - 4: case 1:  $s_1 = \max_{c \in V_{red}} s_1(c)$  (Eq. 4)
  - 5: case 2:  $s_2 = \max_{c \in V_{red}} s_2(c)$  (Eq. 5)
  - 6: case 3: compute  $s_3$  (Eq. 6)
  - 7: re-attach  $v$  according to  $\max(s_1, s_2, s_3)$ .
- 

$$S_{v-in}(c) = \sum_{i \in c} w_{iv} + \sum_{d \in \text{child}(c)} S_{v-in}(d), \quad c \in V_{red} \quad (7)$$

$$S_{v-out}(c) = \sum_{i \in c} w_{vi} + S_{v-out}(p), \quad c \in V_{red} \quad (8)$$

where  $p$  is the parent of  $c$  in  $\mathcal{G}_{red}$ . These recursive definitions allow to compute in linear time  $S_{v-in}(c)$  and  $S_{v-out}(c)$  for all  $c$  (given  $\mathcal{G}_{red}$ ) using dynamic programming, before going over the cases for re-attaching  $v$ .  $S_{v-in}(c)$  is computed going over  $V_{red}$  leaves-to-root (post-order), and  $S_{v-out}(c)$  is computed going over  $V_{red}$  root-to-leaves (pre-order).

Re-attachment is summarized in Algorithm 1. Computing an SCC graph is linear (Cormen et al., 2002) and it is easy to verify that transitive reduction in FRGs is also linear (Line 1). Computing  $S_{v-in}(c)$  and  $S_{v-out}(c)$  (Lines 2-3) is also linear, as explained. Cases 1 and 3 are trivially linear and in case 2 we go over the children of all nodes in  $V_{red}$ . As the reduced graph is a forest, this simply means going over all nodes of  $V_{red}$ , and so the entire algorithm is linear.

Since re-attachment is linear, re-attaching all nodes is quadratic. Thus if we bound the number of iterations over all nodes, the overall complexity is quadratic. This is dramatically more efficient and scalable than applying an ILP solver. In Section 5 we ran TNF until convergence and the maximal number of iterations over graph nodes was 8.

## 4.2 Graph-node-fix

Next, we show Graph-Node-Fix (GNF), a similar approximation that employs the same re-attachment strategy but does not assume the graph is an FRG. Thus, re-attachment of a node  $v$  is done with an ILP solver. Nevertheless, the ILP in GNF is simpler than (1), since we consider only candidate edges

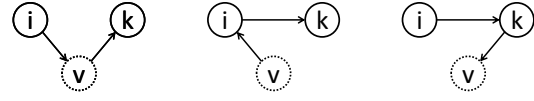


Figure 4: Three types of transitivity constraint violations.

involving  $v$ . Figure 4 illustrates the three types of possible transitivity constraint violations when re-attaching  $v$ . The left side depicts a violation when  $(i, k) \notin E$ , expressed by the constraint in (9) below, and the middle and right depict two violations when the edge  $(i, k) \in E$ , expressed by the constraints in (10). Thus, the ILP is formulated by adding the following constraints to the objective function (3):

$$\forall_{i,k \in V \setminus \{v\}} \text{ if } (i, k) \notin E, \quad x_{iv} + x_{vk} \leq 1 \quad (9)$$

$$\text{ if } (i, k) \in E, \quad x_{vi} \leq x_{vk}, \quad x_{kv} \leq x_{iv} \quad (10)$$

$$x_{iv}, x_{vk} \in \{0, 1\} \quad (11)$$

Complexity is exponential due to the ILP solver; however, the ILP size is reduced by an order of magnitude to  $O(|V|)$  variables and  $O(|V|^2)$  constraints.

## 4.3 Adding local constraints

For some pairs of predicates  $i, j$  we sometimes have prior knowledge whether  $i$  entails  $j$  or not. We term such pairs *local constraints*, and incorporate them into the aforementioned algorithms in the following way. In all algorithms that apply an ILP solver, we add a constraint  $x_{ij} = 1$  if  $i$  entails  $j$  or  $x_{ij} = 0$  if  $i$  does not entail  $j$ . Similarly, in TNF we incorporate local constraints by setting  $w_{ij} = \infty$  or  $w_{ij} = -\infty$ .

## 5 Experiments and Results

In this section we empirically demonstrate that TNF is more efficient than other baselines and its output quality is close to that given by the optimal solution.

### 5.1 Experimental setting

In our experiments we utilize the data set released by Berant et al. (2011). The data set contains 10 entailment graphs, where graph nodes are *typed predicates*. A typed predicate (e.g., ‘ $X_{disease} occur in Y_{country}$ ’) includes a predicate and two *typed variables* that specify the semantic type of the arguments. For instance, the typed variable  $X_{disease}$  can be instantiated by arguments such as ‘*flu*’ or ‘*diabetes*’. The data set contains 39,012 potential edges,

of which 3,427 are annotated as edges (valid entailment rules) and 35,585 are annotated as non-edges.

The data set also contains, for every pair of predicates  $i, j$  in every graph, a *local score*  $s_{ij}$ , which is the output of a classifier trained over distributional similarity features. A positive  $s_{ij}$  indicates that the classifier believes  $i \rightarrow j$ . The weighting function for the graph edges  $w$  is defined as  $w_{ij} = s_{ij} - \lambda$ , where  $\lambda$  is a single parameter controlling graph sparseness: as  $\lambda$  increases,  $w_{ij}$  decreases and becomes negative for more pairs of predicates, rendering the graph more sparse. In addition, the data set contains a set of *local constraints* (see Section 4.3).

We implemented the following algorithms for learning graph edges, where in all of them the graph is first decomposed into components according to Berant et al’s method, as explained in Section 2.

**No-trans** Local scores are used without transitivity constraints – an edge  $(i, j)$  is inserted iff  $w_{ij} > 0$ .

**Exact-graph** Berant et al.’s exact method (2011) for Max-Trans-Graph, which utilizes an ILP solver<sup>1</sup>.

**Exact-forest** Solving Max-Trans-Forest exactly by applying an ILP solver (see Eq. 2).

**LP-relax** Solving Max-Trans-Graph approximately by applying LP-relaxation (see Section 2) on each graph component. We apply the LP solver within the same cutting-plane procedure as Exact-graph to allow for a direct comparison. This also keeps memory consumption manageable, as otherwise all  $|V|^3$  constraints must be explicitly encoded into the LP. As mentioned, our goal is to present a method for learning transitive graphs, while LP-relax produces solutions that violate transitivity. However, we run it on our data set to obtain empirical results, and to compare run-times against TNF.

**Graph-Node-Fix (GNF)** Initialization of each component is performed in the following way: if the graph is very sparse, i.e.  $\lambda \geq C$  for some constant  $C$  (set to 1 in our experiments), then solving the graph exactly is not an issue and we use Exact-graph. Otherwise, we initialize by applying Exact-graph in a sparse configuration, i.e.,  $\lambda = C$ .

**Tree-Node-Fix (TNF)** Initialization is done as in GNF, except that if it generates a graph that is not an FRG, it is corrected by a simple heuristic: for every node in the reduced graph  $\mathcal{G}_{red}$  that has more than

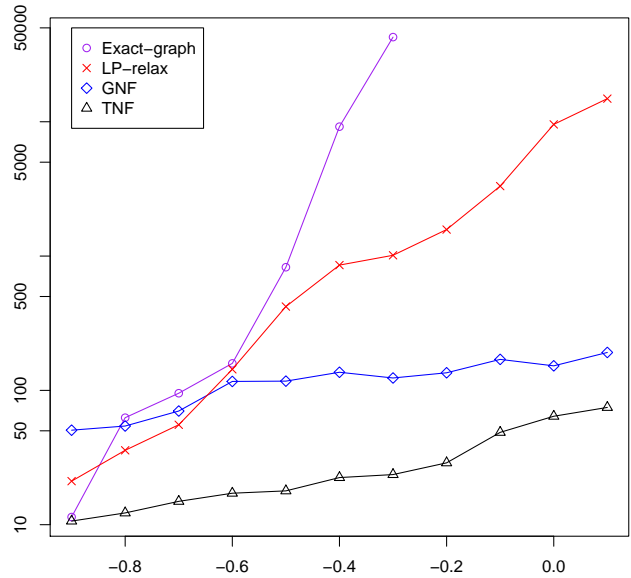


Figure 5: Run-time in seconds for various  $-\lambda$  values.

one parent, we choose from its current parents the single one whose SCC is composed of the largest number of nodes in  $\mathcal{G}$ .

We evaluate algorithms by comparing the set of gold standard edges with the set of edges learned by each algorithm. We measure recall, precision and  $F_1$  for various values of the sparseness parameter  $\lambda$ , and compute the area under the precision-recall Curve (AUC) generated. Efficiency is evaluated by comparing run-times.

## 5.2 Results

We first focus on run-times and show that TNF is efficient and has potential to scale to large data sets.

Figure 5 compares run-times<sup>2</sup> of Exact-graph, GNF, TNF, and LP-relax as  $-\lambda$  increases and the graph becomes denser. Note that the y-axis is in logarithmic scale. Clearly, Exact-graph is extremely slow and run-time increases quickly. For  $\lambda = 0.3$  run-time was already 12 hours and we were unable to obtain results for  $\lambda < 0.3$ , while in TNF we easily got a solution for any  $\lambda$ . When  $\lambda = 0.6$ , where both Exact-graph and TNF achieve best  $F_1$ , TNF is 10 times faster than Exact-graph. When  $\lambda = 0.5$ , TNF is 50 times faster than Exact-graph and so on. Most importantly, run-time for GNF and TNF increases much more slowly than for Exact-graph.

<sup>1</sup>We use the Gurobi optimization package in all experiments.

<sup>2</sup>Run on a multi-core 2.5GHz server with 32GB of RAM.

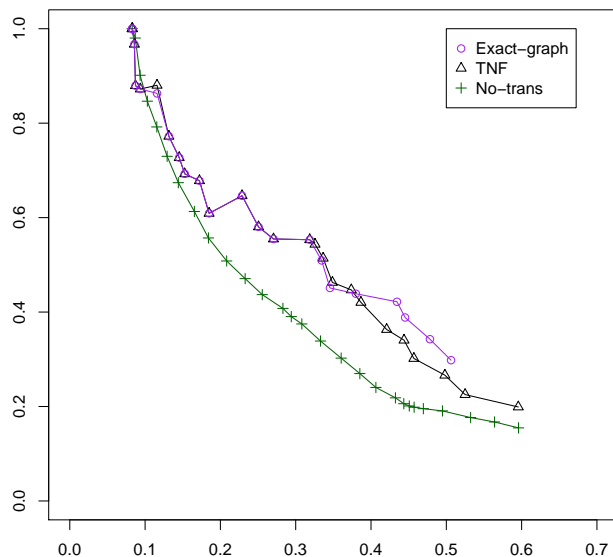


Figure 6: Precision (y-axis) vs. recall (x-axis) curve. Maximal  $F_1$  on the curve is .43 for Exact-graph, .41 for TNF, and .34 for No-trans. AUC in the recall range 0-0.5 is .32 for Exact-graph, .31 for TNF, and .26 for No-trans.

Run-time of LP-relax is also bad compared to TNF and GNF. Run-time increases more slowly than Exact-graph, but still very fast comparing to TNF. When  $\lambda = 0.6$ , LP-relax is almost 10 times slower than TNF, and when  $\lambda = -0.1$ , LP-relax is 200 times slower than TNF. This points to the difficulty of scaling LP-relax to large graphs.

As for the quality of learned graphs, Figure 6 provides a precision-recall curve for Exact-graph, TNF and No-trans (GNF and LP-relax are omitted from the figure and described below to improve readability). We observe that both Exact-graph and TNF substantially outperform No-trans and that TNF’s graph quality is only slightly lower than Exact-graph (which is extremely slow). Following Berant et al., we report in the caption the maximal  $F_1$  on the curve and AUC in the recall range 0-0.5 (the widest range for which we have results for all algorithms). Note that compared to Exact-graph, TNF reduces AUC by a point and the maximal  $F_1$  score by 2 points only.

GNF results are almost identical to those of TNF (maximal  $F_1=0.41$ , AUC: 0.31), and in fact for all  $\lambda$  configurations TNF outperforms GNF by no more than one  $F_1$  point. As for LP-relax, results are just slightly lower than Exact-graph (maximal  $F_1$ : 0.43, AUC: 0.32), but its output is not a transitive graph,

and as shown above run-time is quite slow. Last, we note that the results of Exact-forest are almost identical to Exact-graph (maximal  $F_1$ : 0.43), illustrating that assuming that entailment graphs are FRGs (Section 3) is reasonable in this data set.

To conclude, TNF learns transitive entailment graphs of good quality much faster than Exact-graph. Our experiment utilized an available data set of moderate size; However, we expect TNF to scale to large data sets (that are currently unavailable), where other baselines would be impractical.

## 6 Conclusion

Learning large and accurate resources of entailment rules is essential in many semantic inference applications. Employing transitivity has been shown to improve rule learning, but raises issues of efficiency and scalability.

The first contribution of this paper is a novel modeling assumption that entailment graphs are very similar to FRGs, which is analyzed and validated empirically. The main contribution of the paper is an efficient polynomial approximation algorithm for learning entailment rules, which is based on this assumption. We demonstrate empirically that our method is by orders of magnitude faster than the state-of-the-art exact algorithm, but still produces an output that is almost as good as the optimal solution.

We suggest our method as an important step towards scalable acquisition of precise entailment resources. In future work, we aim to evaluate TNF on large graphs that are automatically generated from huge corpora. This of course requires substantial efforts of pre-processing and test-set annotation. We also plan to examine the benefit of TNF in learning similar structures, e.g., taxonomies or ontologies.

## Acknowledgments

This work was partially supported by the Israel Science Foundation grant 1112/08, the PASCAL-2 Network of Excellence of the European Community FP7-ICT-2007-1-216886, and the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT). The first author has carried out this research in partial fulfilment of the requirements for the Ph.D. degree.



## References

- Alfred V. Aho, Michael R. Garey, and Jeffrey D. Ullman. 1972. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137.
- Roni Ben Aharon, Idan Szpektor, and Ido Dagan. 2010. Generating entailment rules from framenet. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.
- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2010. Global learning of focused entailment graphs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.
- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*.
- Coyne Bob and Owen Rambow. 2009. Lexpar: A freely available english paraphrase lexicon automatically extracted from framenet. In *Proceedings of IEEE International Conference on Semantic Computing*.
- Timothy Chklovski and Patrick Pantel. 2004. Verb ocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of Empirical Methods in Natural Language Processing*.
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2002. *Introduction to Algorithms*. The MIT Press.
- Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2009. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 15(4):1–17.
- Quang Do and Dan Roth. 2010. Constraints based taxonomic relation classification. In *Proceedings of Empirical Methods in Natural Language Processing*.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of Empirical Methods in Natural Language Processing*.
- J. R. Finkel and C. D. Manning. 2008. Enforcing transitivity in coreference resolution. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*.
- Michael R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.
- Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4):343–360.
- Xiao Ling and Dan S. Weld. 2010. Temporal information extraction. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*.
- Andre Martins, Noah Smith, and Eric Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics*.
- Hoifung Poon and Pedro Domingos. 2010. Unsupervised ontology induction from text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.
- Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.
- Sebastian Riedel and James Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proceedings of Empirical Methods in Natural Language Processing*.
- Stefan Schoenmackers, Jesse Davis, Oren Etzioni, and Daniel S. Weld. 2010. Learning first-order horn clauses from web text. In *Proceedings of Empirical Methods in Natural Language Processing*.
- Satoshi Sekine. 2005. Automatic paraphrase discovery based on context and keywords between ne pairs. In *Proceedings of IWP*.
- Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*.
- Rion Snow, Dan Jurafsky, and Andrew Y. Ng. 2006. Semantic taxonomy induction from heterogeneous evidence. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics*.
- Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics*.
- Idan Szpektor and Ido Dagan. 2009. Augmenting wordnet-based inference with argument mapping. In *Proceedings of TextInfer*.
- Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of Empirical Methods in Natural Language Processing*.
- Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research*, 34:255–296.

# Learning High-Level Planning from Text

S.R.K. Branavan, Nate Kushman, Tao Lei, Regina Barzilay

Computer Science and Artificial Intelligence Laboratory

Massachusetts Institute of Technology

{branavan, nkushman, taolei, regina}@csail.mit.edu

## Abstract

Comprehending action preconditions and effects is an essential step in modeling the dynamics of the world. In this paper, we express the semantics of precondition relations extracted from text in terms of planning operations. The challenge of modeling this connection is to ground language at the level of relations. This type of grounding enables us to create high-level plans based on language abstractions. Our model jointly learns to predict precondition relations from text and to perform high-level planning guided by those relations. We implement this idea in the reinforcement learning framework using feedback automatically obtained from plan execution attempts. When applied to a complex virtual world and text describing that world, our relation extraction technique performs on par with a supervised baseline, yielding an F-measure of 66% compared to the baseline’s 65%. Additionally, we show that a high-level planner utilizing these extracted relations significantly outperforms a strong, text unaware baseline – successfully completing 80% of planning tasks as compared to 69% for the baseline.<sup>1</sup>

## 1 Introduction

Understanding action preconditions and effects is a basic step in modeling the dynamics of the world. For example, having *seeds* is a precondition for growing *wheat*. Not surprisingly, preconditions have been extensively explored in various sub-fields of AI. However, existing work on action models has largely focused on tasks and techniques specific to individual sub-fields with little or no interconnection between them. In NLP, precondition relations have been studied in terms of the linguistic mechanisms

<sup>1</sup>The code, data and experimental setup for this work are available at <http://groups.csail.mit.edu/rbg/code/planning>

A pickaxe, which is used to harvest stone, can be made from wood.

(a)

Low Level Actions for: wood → pickaxe → stone

step 1:	move from (0,0) to (2,0)
step 2:	chop tree at: (2,0)
step 3:	get wood at: (2,0)
step 4:	craft plank from wood
step 5:	craft stick from plank
step 6:	craft pickaxe from plank and stick
...	
step N-1:	pickup tool: pickaxe
step N:	harvest stone with pickaxe at: (5,5)

(b)

Figure 1: Text description of preconditions and effects (a), and the low-level actions connecting them (b).

that realize them, while in classical planning, these relations are viewed as a part of world dynamics. In this paper, we bring these two parallel views together, grounding the linguistic realization of these relations in the semantics of planning operations.

The challenge and opportunity of this fusion comes from the mismatch between the abstractions of human language and the granularity of planning primitives. Consider, for example, text describing a virtual world such as Minecraft<sup>2</sup> and a formal description of that world using planning primitives. Due to the mismatch in granularity, even the simple relations between *wood*, *pickaxe* and *stone* described in the sentence in Figure 1a results in dozens of low-level planning actions in the world, as can be seen in Figure 1b. While the text provides a high-level description of world dynamics, it does not provide sufficient details for successful plan execution. On the other hand, planning with low-level actions does not suffer from this limitation, but is computationally intractable for even moderately complex tasks. As a consequence, in many practical domains, planning algorithms rely on manually-crafted high-level

<sup>2</sup><http://www.minecraft.net/>

abstractions to make search tractable (Ghallab et al., 2004; Lekavý and Návrat, 2007).

The central idea of our work is to express the semantics of precondition relations extracted from text in terms of planning operations. For instance, the precondition relation between pickaxe and stone described in the sentence in Figure 1a indicates that plans which involve obtaining stone will likely need to first obtain a pickaxe. The novel challenge of this view is to model grounding at the level of relations, in contrast to prior work which focused on object-level grounding. We build on the intuition that the validity of precondition relations extracted from text can be informed by the execution of a low-level planner.<sup>3</sup> This feedback can enable us to learn these relations without annotations. Moreover, we can use the learned relations to guide a high level planner and ultimately improve planning performance.

We implement these ideas in the reinforcement learning framework, wherein our model jointly learns to predict precondition relations from text and to perform high-level planning guided by those relations. For a given planning task and a set of candidate relations, our model repeatedly predicts a sequence of subgoals where each subgoal specifies an attribute of the world that must be made true. It then asks the low-level planner to find a plan between each consecutive pair of subgoals in the sequence. The observed feedback – whether the low-level planner succeeded or failed at each step – is utilized to update the policy for both text analysis and high-level planning.

We evaluate our algorithm in the Minecraft virtual world, using a large collection of user-generated online documents as our source of textual information. Our results demonstrate the strength of our relation extraction technique – while using planning feedback as its only source of supervision, it achieves a precondition relation extraction accuracy on par with that of a supervised SVM baseline. Specifically, it yields an F-score of 66% compared to the 65% of the baseline. In addition, we show that these extracted relations can be used to improve the performance of a high-level planner. As baselines

<sup>3</sup>If a planner can find a plan to successfully obtain stone after obtaining a pickaxe, then a pickaxe is likely a precondition for stone. Conversely, if a planner obtains stone without first obtaining a pickaxe, then it is likely not a precondition.

for this evaluation, we employ the Metric-FF planner (Hoffmann and Nebel, 2001),<sup>4</sup> as well as a text-unaware variant of our model. Our results show that our text-driven high-level planner significantly outperforms all baselines in terms of completed planning tasks – it successfully solves 80% as compared to 41% for the Metric-FF planner and 69% for the text unaware variant of our model. In fact, the performance of our method approaches that of an oracle planner which uses manually-annotated preconditions.

## 2 Related Work

**Extracting Event Semantics from Text** The task of extracting preconditions and effects has previously been addressed in the context of lexical semantics (Sil et al., 2010; Sil and Yates, 2011). These approaches combine large-scale distributional techniques with supervised learning to identify desired semantic relations in text. Such combined approaches have also been shown to be effective for identifying other relationships between events, such as causality (Girju and Moldovan, 2002; Chang and Choi, 2006; Blanco et al., 2008; Beamer and Girju, 2009; Do et al., 2011).

Similar to these methods, our algorithm capitalizes on surface linguistic cues to learn preconditions from text. However, our only source of supervision is the feedback provided by the planning task which utilizes the predictions. Additionally, we not only identify these relations in text, but also show they are valuable in performing an external task.

**Learning Semantics via Language Grounding** Our work fits into the broad area of grounded language acquisition, where the goal is to learn linguistic analysis from a situated context (Oates, 2001; Siskind, 2001; Yu and Ballard, 2004; Fleischman and Roy, 2005; Mooney, 2008a; Mooney, 2008b; Branavan et al., 2009; Liang et al., 2009; Vogel and Jurafsky, 2010). Within this line of work, we are most closely related to the reinforcement learning approaches that learn language by interacting with an external environment (Branavan et al., 2009; Branavan et al., 2010; Vogel and Jurafsky, 2010; Branavan et al., 2011).

<sup>4</sup>The state-of-the-art baseline used in the 2008 International Planning Competition. <http://ipc.informatik.uni-freiburg.de/>

#### Text (input):

A **pickaxe**, which is used to harvest **stone**, can be made from **wood**.

#### Precondition Relations:



#### Plan Subgoal Sequence:

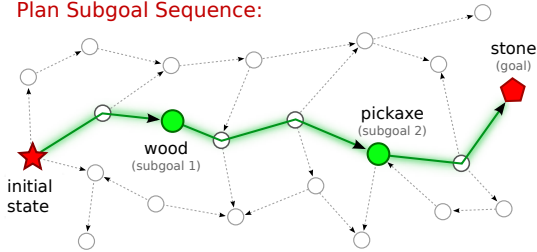


Figure 2: A high-level plan showing two subgoals in a precondition relation. The corresponding sentence is shown above.

The key distinction of our work is the use of grounding to learn abstract pragmatic relations, i.e. to learn linguistic patterns that describe relationships between objects in the world. This supplements previous work which grounds words to objects in the world (Branavan et al., 2009; Vogel and Jurafsky, 2010). Another important difference of our setup is the way the textual information is utilized in the situated context. Instead of getting step-by-step instructions from the text, our model uses text that describes general knowledge about the domain structure. From this text, it extracts relations between objects in the world which hold independently of any given task. Task-specific solutions are then constructed by a planner that relies on these relations to perform effective high-level planning.

**Hierarchical Planning** It is widely accepted that high-level plans that factorize a planning problem can greatly reduce the corresponding search space (Newell et al., 1959; Bacchus and Yang, 1994). Previous work in planning has studied the theoretical properties of valid abstractions and proposed a number of techniques for generating them (Jonsson and Barto, 2005; Wolfe and Barto, 2005; Mehta et al., 2008; Barry et al., 2011). In general, these techniques use static analysis of the low-level domain to induce effective high-level abstractions. In contrast, our focus is on learning the abstraction from natural language. Thus our technique is complementary to past work, and can benefit from human knowledge about the domain structure.

### 3 Problem Formulation

Our task is two-fold. First, given a text document describing an environment, we wish to extract a set of precondition/effect relations implied by the text. Second, we wish to use these induced relations to determine an action sequence for completing a given task in the environment.

We formalize our task as illustrated in Figure 2. As input, we are given a world defined by the tuple  $\langle S, A, T \rangle$ , where  $S$  is the set of possible world states,  $A$  is the set of possible actions and  $T$  is a deterministic state transition function. Executing action  $a$  in state  $s$  causes a transition to a new state  $s'$  according to  $T(s' | s, a)$ . States are represented using propositional logic predicates  $x_i \in X$ , where each state is simply a set of such predicates, i.e.  $s \subset X$ .

The objective of the text analysis part of our task is to automatically extract a set of valid precondition/effect relationships from a given document  $d$ . Given our definition of the world state, preconditions and effects are merely single term predicates,  $x_i$ , in this world state. We assume that we are given a seed mapping between a predicate  $x_i$ , and the word types in the document that reference it (see Table 3 for examples). Thus, for each predicate pair  $\langle x_k, x_l \rangle$ , we want to utilize the text to predict whether  $x_k$  is a precondition for  $x_l$ ; i.e.,  $x_k \rightarrow x_l$ . For example, from the text in Figure 2, we want to predict that possessing a pickaxe is a precondition for possessing stone. Note that this relation implies the reverse as well, i.e.  $x_l$  can be interpreted as the effect of an action sequence performed on state  $x_k$ .

Each planning goal  $g \in G$  is defined by a starting state  $s_0^g$ , and a final goal state  $s_f^g$ . This goal state is represented by a set of predicates which need to be made true. In the planning part of our task our objective is to find a sequence of actions  $\vec{a}$  that connect  $s_0^g$  to  $s_f^g$ . Finally, we assume document  $d$  does not contain step-by-step instructions for any individual task, but instead describes general facts about the given world that are useful for a wide variety of tasks.

### 4 Model

The key idea behind our model is to leverage textual descriptions of preconditions and effects to guide the construction of high level plans. We define a high-level plan as a sequence of *subgoals*, where each

subgoal is represented by a single-term predicate,  $x_i$ , that needs to be set in the corresponding world state – e.g. `have(wheat)=true`. Thus the set of possible subgoals is defined by the set of all possible single-term predicates in the domain. In contrast to low-level plans, the transition between these subgoals can involve multiple low-level actions. Our algorithm for textually informed high-level planning operates in four steps:

1. Use text to predict the preconditions of each subgoal. These predictions are for the entire domain and are not goal specific.
2. Given a planning goal and the induced preconditions, predict a subgoal sequence that achieves the given goal.
3. Execute the predicted sequence by giving each pair of consecutive subgoals to a low-level planner. This planner, treated as a black-box, computes the low-level plan actions necessary to transition from one subgoal to the next.
4. Update the model parameters, using the low-level planner’s success or failure as the source of supervision.

We formally define these steps below.

**Modeling Precondition Relations** Given a document  $d$ , and a set of subgoal pairs  $\langle x_i, x_j \rangle$ , we want to predict whether subgoal  $x_i$  is a precondition for  $x_j$ . We assume that precondition relations are generally described within single sentences. We first use our seed grounding in a preprocessing step where we extract all predicate pairs where both predicates are mentioned in the same sentence. We call this set the *Candidate Relations*. Note that this set will contain many invalid relations since co-occurrence in a sentence does not necessarily imply a valid precondition relation.<sup>5</sup> Thus for each sentence,  $\vec{w}_k$ , associated with a given *Candidate Relation*,  $x_i \rightarrow x_j$ , our task is to predict whether the sentence indicates the relation. We model this decision via a log linear distribution as follows:

$$p(x_i \rightarrow x_j \mid \vec{w}_k, q_k; \theta_c) \propto e^{\theta_c \cdot \phi_c(x_i, x_j, \vec{w}_k, q_k)}, \quad (1)$$

where  $\theta_c$  is the vector of model parameters. We compute the feature function  $\phi_c$  using the seed

<sup>5</sup>In our dataset only 11% of *Candidate Relations* are valid.

**Input:** A document  $d$ , Set of planning tasks  $G$ ,  
Set of candidate precondition relations  $C_{all}$ ,  
Reward function  $r()$ , Number of iterations  $T$

**Initialization:** Model parameters  $\theta_x = 0$  and  $\theta_c = 0$ .

```

for  $i = 1 \dots T$  do
  Sample valid preconditions:
   $C \leftarrow \emptyset$ 
  foreach  $\langle x_i, x_j \rangle \in C_{all}$  do
    foreach Sentence  $\vec{w}_k$  containing  $x_i$  and  $x_j$  do
       $v \sim p(x_i \rightarrow x_j \mid \vec{w}_k, q_k; \theta_c)$ 
      if  $v = 1$  then  $C = C \cup \langle x_i, x_j \rangle$ 
    end
  end
  Predict subgoal sequences for each task g.
  foreach  $g \in G$  do
    Sample subgoal sequence  $\vec{x}$  as follows:
    for  $t = 1 \dots n$  do
      Sample next subgoal:
       $x_t \sim p(x \mid x_{t-1}, s_0^g, s_f^g, C; \theta_x)$ 
      Construct low-level subtask from  $x_{t-1}$  to  $x_t$ 
      Execute low-level planner on subtask
    end
    Update subgoal prediction model using Eqn. 2
  end
  Update text precondition model using Eqn. 3
end

```

Algorithm 1: A policy gradient algorithm for parameter estimation in our model.

grounding, the sentence  $\vec{w}_k$ , and a given dependency parse  $q_k$  of the sentence. Given these per-sentence decisions, we predict the set of all valid precondition relations,  $C$ , in a deterministic fashion. We do this by considering a precondition  $x_i \rightarrow x_j$  as valid if it is predicted to be valid by at least one sentence.

**Modeling Subgoal Sequences** Given a planning goal  $g$ , defined by initial and final goal states  $s_0^g$  and  $s_f^g$ , our task is to predict a sequence of subgoals  $\vec{x}$  which will achieve the goal. We condition this decision on our predicted set of valid preconditions  $C$ , by modeling the distribution over sequences  $\vec{x}$  as:

$$p(\vec{x} \mid s_0^g, s_f^g, C; \theta_x) = \prod_{t=1}^n p(x_t \mid x_{t-1}, s_0^g, s_f^g, C; \theta_x),$$

$$p(x_t \mid x_{t-1}, s_0^g, s_f^g, C; \theta_x) \propto e^{\theta_x \cdot \phi_x(x_t, x_{t-1}, s_0^g, s_f^g, C)}.$$

Here we assume that subgoal sequences are Markovian in nature and model individual subgoal predictions using a log-linear model. Note that in con-

trast to Equation 1 where the predictions are goal-agnostic, these predictions are goal-specific. As before,  $\theta_x$  is the vector of model parameters, and  $\phi_x$  is the feature function. Additionally, we assume a special stop symbol,  $x_\emptyset$ , which indicates the end of the subgoal sequence.

**Parameter Update** Parameter updates in our model are done via reinforcement learning. Specifically, once the model has predicted a subgoal sequence for a given goal, the sequence is given to the low-level planner for execution. The success or failure of this execution is used to compute the reward signal  $r$  for parameter estimation. This predict-execute-update cycle is repeated until convergence. We assume that our reward signal  $r$  strongly correlates with the correctness of model predictions. Therefore, during learning, we need to find the model parameters that maximize expected future reward (Sutton and Barto, 1998). We perform this maximization via stochastic gradient ascent, using the standard policy gradient algorithm (Williams, 1992; Sutton et al., 2000).

We perform two separate policy gradient updates, one for each model component. The objective of the text component of our model is purely to predict the validity of preconditions. Therefore, subgoal pairs  $\langle x_k, x_l \rangle$ , where  $x_l$  is reachable from  $x_k$ , are given positive reward. The corresponding parameter update, with learning rate  $\alpha_c$ , takes the following form:

$$\Delta\theta_c \leftarrow \alpha_c r \left[ \phi_c(x_i, x_j, \vec{w}_k, q_k) - \mathbb{E}_{p(x_{i'} \rightarrow x_{j'} | \cdot)} \left[ \phi_c(x_{i'}, x_{j'}, \vec{w}_k, q_k) \right] \right]. \quad (2)$$

The objective of the planning component of our model is to predict subgoal sequences that successfully achieve the given planning goals. Thus we directly use plan-success as a binary reward signal, which is applied to each subgoal decision in a sequence. This results in the following update:

$$\Delta\theta_x \leftarrow \alpha_x r \sum_t \left[ \phi_x(x_t, x_{t-1}, s_0^g, s_f^g, C) - \mathbb{E}_{p(x'_t | \cdot)} \left[ \phi_x(x'_t, x_{t-1}, s_0^g, s_f^g, C) \right] \right], \quad (3)$$

where  $t$  indexes into the subgoal sequence and  $\alpha_x$  is the learning rate.

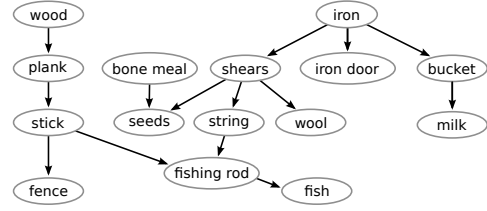


Figure 3: Example of the precondition dependencies present in the Minecraft domain.

Domain	#Objects	#Pred Types	#Actions
Parking	49	5	4
Floortile	61	10	7
Barman	40	15	12
<b>Minecraft</b>	<b>108</b>	<b>16</b>	<b>68</b>

Table 1: A comparison of complexity between Minecraft and some domains used in the IPC-2011 sequential satisficing track. In the Minecraft domain, the number of objects, predicate types, and actions is significantly larger.

## 5 Applying the Model

We apply our method to Minecraft, a grid-based virtual world. Each grid location represents a tile of either land or water and may also contain resources. Users can freely move around the world, harvest resources and craft various tools and objects from these resources. The dynamics of the world require certain resources or tools as prerequisites for performing a given action, as can be seen in Figure 3. For example, a user must first craft a *bucket* before they can collect *milk*.

**Defining the Domain** In order to execute a traditional planner on the Minecraft domain, we define the domain using the Planning Domain Definition Language (PDDL) (Fox and Long, 2003). This is the standard task definition language used in the International Planning Competitions (IPC).<sup>6</sup> We define as predicates all aspects of the game state – for example, the location of resources in the world, the resources and objects possessed by the player, and the player’s location. Our subgoals  $x_i$  and our task goals  $s_f^g$  map directly to these predicates. This results in a domain with significantly greater complexity than those solvable by traditional low-level planners. Table 1 compares the complexity of our domain with some typical planning domains used in the IPC.

<sup>6</sup><http://ipc.icaps-conference.org/>

**Low-level Planner** As our low-level planner we employ Metric-FF (Hoffmann and Nebel, 2001), the state-of-the-art baseline used in the 2008 International Planning Competition. Metric-FF is a forward-chaining heuristic state space planner. Its main heuristic is to simplify the task by ignoring operator delete lists. The number of actions in the solution for this simplified task is then used as the goal distance estimate for various search strategies.

**Features** The two components of our model leverage different types of information, and as a result, they each use distinct sets of features. The text component features  $\phi_c$  are computed over sentences and their dependency parses. The Stanford parser (de Marneffe et al., 2006) was used to generate the dependency parse information for each sentence. Examples of these features appear in Table 2. The sequence prediction component takes as input both the preconditions induced by the text component as well as the planning state and the previous subgoal. Thus  $\phi_x$  contains features which check whether two subgoals are connected via an induced precondition relation, in addition to features which are simply the Cartesian product of domain predicates.

## 6 Experimental Setup

**Datasets** As the text description of our virtual world, we use documents from the Minecraft Wiki,<sup>7</sup> the most popular information source about the game. Our manually constructed seed grounding of predicates contains 74 entries, examples of which can be seen in Table 3. We use this seed grounding to identify a set of 242 sentences that reference predicates in the Minecraft domain. This results in a set of 694 *Candidate Relations*. We also manually annotated the relations expressed in the text, identifying 94 of the *Candidate Relations* as valid. Our corpus contains 979 unique word types and is composed of sentences with an average length of 20 words.

We test our system on a set of 98 problems that involve collecting resources and constructing objects in the Minecraft domain – for example, fishing, cooking and making furniture. To assess the complexity of these tasks, we manually constructed high-level plans for these goals and solved them using the Metric-FF planner. On average, the execu-

Words
Dependency Types
Dependency Type $\times$ Direction
Word $\times$ Dependency Type
Word $\times$ Dependency Type $\times$ Direction

Table 2: Example text features. A subgoal pair  $\langle x_i, x_j \rangle$  is first mapped to word tokens using a small grounding table. Words and dependencies are extracted along paths between mapped target words. These are combined with path directions to generate the text features.

Domain Predicate	Noun Phrases
have(plank)	wooden plank, wood plank
have(stone)	stone, cobblestone
have(iron)	iron ingot

Table 3: Examples in our seed grounding table. Each predicate is mapped to one or more noun phrases that describe it in the text.

tion of the sequence of low-level plans takes 35 actions, with 3 actions for the shortest plan and 123 actions for the longest. The average branching factor is 9.7, leading to an average search space of more than  $10^{34}$  possible action sequences. For evaluation purposes we manually identify a set of *Gold Relations* consisting of all precondition relations that are valid in this domain, including those not discussed in the text.

**Evaluation Metrics** We use our manual annotations to evaluate the type-level accuracy of relation extraction. To evaluate our high-level planner, we use the standard measure adopted by the IPC. This evaluation measure simply assesses whether the planner completes a task within a predefined time.

**Baselines** To evaluate the performance of our relation extraction, we compare against an SVM classifier<sup>8</sup> trained on the *Gold Relations*. We test the SVM baseline in a leave-one-out fashion.

To evaluate the performance of our text-aware high-level planner, we compare against five baselines. The first two baselines – *FF* and *No Text* – do not use any textual information. The *FF* baseline directly runs the Metric-FF planner on the given task, while the *No Text* baseline is a variant of our model that learns to plan in the reinforcement learning framework. It uses the same state-level features

<sup>7</sup>[http://www.minecraftwiki.net/wiki/Minecraft\\_Wiki/](http://www.minecraftwiki.net/wiki/Minecraft_Wiki/)

<sup>8</sup>SVM<sup>light</sup> (Joachims, 1999) with default parameters.



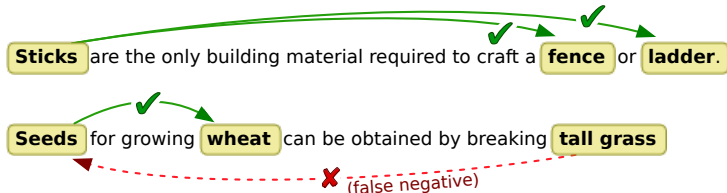


Figure 4: Examples of precondition relations predicted by our model from text. Check marks (✓) indicate correct predictions, while a cross (✗) marks the incorrect one – in this case, a valid relation that was predicted as invalid by our model. Note that each pair of highlighted noun phrases in a sentence is a *Candidate Relation*, and pairs that are not connected by an arrow were correctly predicted to be invalid by our model.

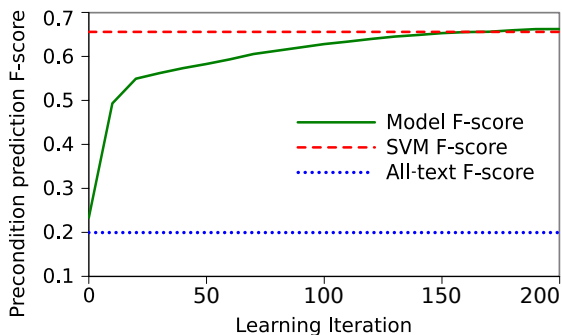


Figure 5: The performance of our model and a supervised SVM baseline on the precondition prediction task. Also shown is the F-Score of the full set of *Candidate Relations* which is used unmodified by *All Text*, and is given as input to our model. Our model’s F-score, averaged over 200 trials, is shown with respect to learning iterations.

as our model, but does not have access to text.

The *All Text* baseline has access to the full set of 694 *Candidate Relations*. During learning, our full model refines this set of relations, while in contrast the *All Text* baseline always uses the full set.

The two remaining baselines constitute the upper bound on the performance of our model. The first, *Manual Text*, is a variant of our model which directly uses the links derived from manual annotations of preconditions in text. The second, *Gold*, has access to the *Gold Relations*. Note that the connections available to *Manual Text* are a subset of the *Gold* links, because the text does not specify all relations.

**Experimental Details** All experimental results are averaged over 200 independent runs for both our model as well as the baselines. Each of these trials is run for 200 learning iterations with a maximum subgoal sequence length of 10. To find a low-level plan between each consecutive pair of subgoals, our high-level planner internally uses Metric-FF. We give Metric-FF a one-minute timeout to find such a low-level plan. To ensure that the comparison

Method	% Plans
FF	40.8
No text	69.4
All text	75.5
<b>Full model</b>	<b>80.2</b>
Manual text	84.7
Gold connection	87.1

Table 4: Percentage of tasks solved successfully by our model and the baselines. All performance differences between methods are statistically significant at  $p \leq .01$ .

between the high-level planners and the FF baseline is fair, the FF baseline is allowed a runtime of 2,000 minutes. This is an upper bound on the time that our high-level planner can take over the 200 learning iterations, with subgoal sequences of length at most 10 and a one minute timeout. Lastly, during learning we initialize all parameters to zero, use a fixed learning rate of 0.0001, and encourage our model to explore the state space by using the standard  $\epsilon$ -greedy exploration strategy (Sutton and Barto, 1998).

## 7 Results

**Relation Extraction** Figure 5 shows the performance of our method on identifying preconditions in text. We also show the performance of the supervised SVM baseline. As can be seen, after 200 learning iterations, our model achieves an F-Measure of 66%, equal to the supervised baseline. These results support our hypothesis that planning feedback is a powerful source of supervision for analyzing a given text corpus. Figure 4 shows some examples of sentences and the corresponding extracted relations.

**Planning Performance** As shown in Table 4 our text-enriched planning model outperforms the text-free baselines by more than 10%. Moreover, the performance improvement of our model over the *All Text* baseline demonstrates that the accuracy of the



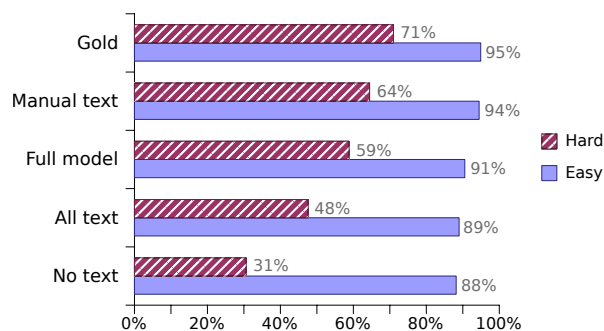


Figure 6: Percentage of problems solved by various models on Easy and Hard problem sets.

extracted text relations does indeed impact planning performance. A similar conclusion can be reached by comparing the performance of our model and the *Manual Text* baseline.

The difference in performance of 2.35% between *Manual Text* and *Gold* shows the importance of the precondition information that is missing from the text. Note that *Gold* itself does not complete all tasks – this is largely because the Markov assumption made by our model does not hold for all tasks.<sup>9</sup>

Figure 6 breaks down the results based on the difficulty of the corresponding planning task. We measure problem complexity in terms of the low-level steps needed to implement a manually constructed high-level plan. Based on this measure, we divide the problems into two sets. As can be seen, all of the high-level planners solve almost all of the easy problems. However, performance varies greatly on the more challenging tasks, directly correlating with planner sophistication. On these tasks our model outperforms the *No Text* baseline by 28% and the *All Text* baseline by 11%.

**Feature Analysis** Figure 7 shows the top five positive features for our model and the SVM baseline. Both models picked up on the words that indicate precondition relations in this domain. For instance, the word *use* often occurs in sentences that describe the resources required to make an object, such as “bricks are items used to craft brick blocks”. In addition to lexical features, dependency information is also given high weight by both learners. An example

<sup>9</sup>When a given task has two non-trivial preconditions, our model will choose to satisfy one of the two first, and the Markov assumption blinds it to the remaining precondition, preventing it from determining that it must still satisfy the other.

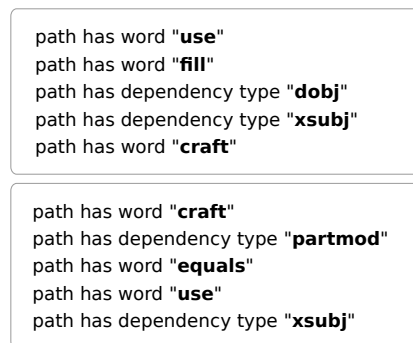


Figure 7: The top five positive features on words and dependency types learned by our model (above) and by SVM (below) for precondition prediction.

of this is a feature that checks for the direct object dependency type. This analysis is consistent with prior work on event semantics which shows lexico-syntactic features are effective cues for learning text relations (Blanco et al., 2008; Beamer and Girju, 2009; Do et al., 2011).

## 8 Conclusions

In this paper, we presented a novel technique for inducing precondition relations from text by grounding them in the semantics of planning operations. While using planning feedback as its only source of supervision, our method for relation extraction achieves a performance on par with that of a supervised baseline. Furthermore, relation grounding provides a new view on classical planning problems which enables us to create high-level plans based on language abstractions. We show that building high-level plans in this manner significantly outperforms traditional techniques in terms of task completion.

## Acknowledgments

The authors acknowledge the support of the NSF (CAREER grant IIS-0448168, grant IIS-0835652), the DARPA Machine Reading Program (FA8750-09-C-0172, PO#4910018860), and Batelle (PO#300662). Thanks to Amir Globerson, Tommi Jaakkola, Leslie Kaelbling, George Konidaris, Dylan Hadfield-Menell, Stefanie Tellex, the MIT NLP group, and the ACL reviewers for their suggestions and comments. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors, and do not necessarily reflect the views of the funding organizations.

## References

- Fahiem Bacchus and Qiang Yang. 1994. Downward refinement and the efficiency of hierarchical problem solving. *Artificial Intell.*, 71(1):43–100.
- Jennifer L. Barry, Leslie Pack Kaelbling, and Toms Lozano-Prez. 2011. DetH\*: Approximate hierarchical solution of large markov decision processes. In *IJCAI'11*, pages 1928–1935.
- Brandon Beamer and Roxana Girju. 2009. Using a bigram event model to predict causal potential. In *Proceedings of CICLing*, pages 430–441.
- Eduardo Blanco, Nuria Castell, and Dan Moldovan. 2008. Causal relation extraction. In *Proceedings of the LREC'08*.
- S.R.K Branavan, Harr Chen, Luke Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proceedings of ACL*, pages 82–90.
- S.R.K Branavan, Luke Zettlemoyer, and Regina Barzilay. 2010. Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of ACL*, pages 1268–1277.
- S. R. K. Branavan, David Silver, and Regina Barzilay. 2011. Learning to win by reading manuals in a monte-carlo framework. In *Proceedings of ACL*, pages 268–277.
- Du-Seong Chang and Key-Sun Choi. 2006. Incremental cue phrase learning and bootstrapping method for causality extraction using cue phrase and word pair probabilities. *Inf. Process. Manage.*, 42(3):662–678.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC 2006*.
- Q. Do, Y. Chan, and D. Roth. 2011. Minimally supervised event causality identification. In *EMNLP*, 7.
- Michael Fleischman and Deb Roy. 2005. Intentional context in situated natural language learning. In *Proceedings of CoNLL*, pages 104–111.
- Maria Fox and Derek Long. 2003. Pddl2.1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:2003.
- Malik Ghallab, Dana S. Nau, and Paolo Traverso. 2004. *Automated Planning: theory and practice*. Morgan Kaufmann.
- Roxana Girju and Dan I. Moldovan. 2002. Text mining for causal relations. In *Proceedings of FLAIRS*, pages 360–364.
- Jörg Hoffmann and Bernhard Nebel. 2001. The FF planning system: Fast plan generation through heuristic search. *JAIR*, 14:253–302.
- Thorsten Joachims. 1999. Advances in kernel methods. chapter Making large-scale support vector machine learning practical, pages 169–184. MIT Press.
- Anders Jonsson and Andrew Barto. 2005. A causal approach to hierarchical decomposition of factored mdps. In *Advances in Neural Information Processing Systems*, 13:10541060, page 22. Press.
- Marián Lekavý and Pavol Návrát. 2007. Expressivity of strips-like and htn-like planning. *Lecture Notes in Artificial Intelligence*, 4496:121–130.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of ACL*, pages 91–99.
- Neville Mehta, Soumya Ray, Prasad Tadepalli, and Thomas Dietterich. 2008. Automatic discovery and transfer of maxq hierarchies. In *Proceedings of the 25th international conference on Machine learning, ICML '08*, pages 648–655.
- Raymond J. Mooney. 2008a. Learning language from its perceptual context. In *Proceedings of ECML/PKDD*.
- Raymond J. Mooney. 2008b. Learning to connect language and perception. In *Proceedings of AAIL*, pages 1598–1601.
- A. Newell, J.C. Shaw, and H.A. Simon. 1959. *The processes of creative thinking*. Paper P-1320. Rand Corporation.
- James Timothy Oates. 2001. *Grounding knowledge in sensors: Unsupervised learning for language and planning*. Ph.D. thesis, University of Massachusetts Amherst.
- Avirup Sil and Alexander Yates. 2011. Extracting STRIPS representations of actions and events. In *Recent Advances in Natural Language Learning (RANLP)*.
- Avirup Sil, Fei Huang, and Alexander Yates. 2010. Extracting action and event semantics from web text. In *AAAI 2010 Fall Symposium on Commonsense Knowledge (CSK)*.
- Jeffrey Mark Siskind. 2001. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of Artificial Intelligence Research*, 15:31–90.
- Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. The MIT Press.
- Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in NIPS*, pages 1057–1063.
- Adam Vogel and Daniel Jurafsky. 2010. Learning to follow navigational directions. In *Proceedings of the ACL*, pages 806–814.
- Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8.

Alicia P. Wolfe and Andrew G. Barto. 2005. Identifying useful subgoals in reinforcement learning by local graph partitioning. In *In Proceedings of the Twenty-Second International Conference on Machine Learning*, pages 816–823.

Chen Yu and Dana H. Ballard. 2004. On the integration of grounding language and learning objects. In *Proceedings of AAAI*, pages 488–493.

# Distributional Semantics in Technicolor

**Elia Bruni**

University of Trento  
elia.bruni@unitn.it

**Gemma Boleda**

University of Texas at Austin  
gemma.boleda@utcompling.com

**Marco Baroni**

**Nam-Khanh Tran**  
University of Trento  
name.surname@unitn.it

## Abstract

Our research aims at building computational models of word meaning that are perceptually grounded. Using computer vision techniques, we build visual and multimodal distributional models and compare them to standard textual models. Our results show that, while visual models with state-of-the-art computer vision techniques perform worse than textual models in general tasks (accounting for semantic relatedness), they are as good or better models of the meaning of words with visual correlates such as color terms, even in a nontrivial task that involves nonliteral uses of such words. Moreover, we show that visual and textual information are tapping on different aspects of meaning, and indeed combining them in multimodal models often improves performance.

## 1 Introduction

Traditional semantic space models represent meaning on the basis of word co-occurrence statistics in large text corpora (Turney and Pantel, 2010). These models (as well as virtually all work in computational lexical semantics) rely on verbal information only, while human semantic knowledge also relies on non-verbal experience and representation (Louwerse, 2011), crucially on the information gathered through perception. Recent developments in computer vision make it possible to computationally model one vital human perceptual channel: vision (Mooney, 2008). A few studies have begun to use visual information extracted from images as part of distributional semantic models (Bergsma and Van

Durme, 2011; Bergsma and Goebel, 2011; Bruni et al., 2011; Feng and Lapata, 2010; Leong and Mihalcea, 2011). These preliminary studies all focus on how vision may help text-based models in general terms, by evaluating performance on, for instance, word similarity datasets such as WordSim353.

This paper contributes to connecting language and perception, focusing on how to exploit visual information to build better models of word meaning, in three ways: (1) We carry out a systematic comparison of models using textual, visual, and both types of information. (2) We evaluate the models on general semantic relatedness tasks and on two specific tasks where visual information is highly relevant, as they focus on color terms. (3) Unlike previous work, we study the impact of using different kinds of visual information for these semantic tasks.

Our results show that, while visual models with state-of-the-art computer vision techniques perform worse than textual models in general semantic tasks, they are as good or better models of the meaning of words with visual correlates such as color terms, even in a nontrivial task that involves nonliteral uses of such words. Moreover, we show that visual and textual information are tapping on different aspects of meaning, such that they are complementary sources of information, and indeed combining them in multimodal models often improves performance. We also show that “hybrid” models exploiting the patterns of co-occurrence of words as tags of the same images can be a powerful surrogate of visual information under certain circumstances.

The rest of the paper is structured as follows. Section 2 introduces the textual, visual, multimodal,

and hybrid models we use for our experiments. We present our experiments in sections 3 to 5. Section 6 reviews related work, and section 7 finishes with conclusions and future work.

## 2 Distributional semantic models

### 2.1 Textual models

For the current project, we constructed a set of textual distributional models that implement various standard ways to extract them from a corpus, chosen to be representative of the state of the art. In all cases, occurrence and co-occurrence statistics are extracted from the freely available ukWaC and Wackypedia corpora combined (size: 1.9B and 820M tokens, respectively).<sup>1</sup> Moreover, in all models the raw co-occurrence counts are transformed into nonnegative Local Mutual Information (LMI) scores.<sup>2</sup> Finally, in all models we harvest vector representations for the same words (lemmas), namely the top 20K most frequent nouns, 5K most frequent adjectives and 5K most frequent verbs in the combined corpora (for coherence with the vision-based models, that cannot exploit contextual information to distinguish nouns and adjectives, we merge nominal and adjectival usages of the color adjectives in the text-based models as well). The same 30K target nouns, verbs and adjectives are also employed as contextual elements.

The **Window2** and **Window20** models are based on counting co-occurrences with collocates within a window of fixed width, in the tradition of HAL (Lund and Burgess, 1996). Window2 records sentence-internal co-occurrence with the nearest 2 content words to the left and right of each target concept, a narrow context definition expected to capture taxonomic relations. Window20 considers a larger window of 20 words to the left and right of the target, and should capture broader topical relations. The **Document** model corresponds to a “topic-based” approach in which words are represented as distributions over documents. It is based on a word-by-document matrix, recording the distribution of the

<sup>1</sup><http://wacky.sslmit.unibo.it/>

<sup>2</sup>LMI is obtained by multiplying raw counts by Pointwise Mutual Information, and it is a close approximation to the Log-Likelihood Ratio (Evert, 2005). It counteracts the tendency of PMI to favour extremely rare events.

30K target words across the 30K documents in the concatenated corpus that have the largest cumulative LMI mass. This model is thus akin to traditional Latent Semantic Analysis (Landauer and Dumais, 1997), without dimensionality reduction.

We add to the models we constructed the freely available Distributional Memory (**DM**) model,<sup>3</sup> that has been shown to reach state-of-the-art performance in many semantic tasks (Baroni and Lenci, 2010). DM is an example of a more complex text-based model that exploits lexico-syntactic and dependency relations between words (see Baroni and Lenci’s article for details), and we use it as an instance of a grammar-based model. DM is based on the same corpora we used plus the 100M-word British National Corpus,<sup>4</sup> and it also uses LMI scores.

### 2.2 Visual models

The visual models use information extracted from images instead of textual corpora. We use image data where each image is associated with one or more words or **tags** (we use “tag” for each word associated to the image, and “label” for the set of tags of an image). We use the ESP-Game dataset,<sup>5</sup> containing 100K images labeled through a game with a purpose in which two people partnered online must independently and rapidly agree on an appropriate word to label randomly selected images. Once a word is entered by both partners in a certain number of game matches, that word is added to the label for that image, and it becomes a taboo word for the following rounds of the game (von Ahn and Dabbish, 2004). There are 20,515 distinct tags in the dataset, with an average of 4 tags per image. We build one vector with visual features for each tag in the dataset.

The visual features are extracted with the use of a standard bag-of-visual-words (**BoVW**) representation of images, inspired by NLP (Sivic and Zisserman, 2003; Csurka et al., 2004; Nister and Stewenius, 2006; Bosch et al., 2007; Yang et al., 2007). This approach relies on the notion of a common vocabulary of “visual words” that can serve as discrete representations for all images. Contrary to what hap-

<sup>3</sup><http://cllc.cimec.unitn.it/dm>

<sup>4</sup><http://www.natcorp.ox.ac.uk/>

<sup>5</sup><http://www.espgame.org>

pens in NLP, where words are (mostly) discrete and easy to identify, in vision the visual words need to be first defined. The process is completely inductive. In a nutshell, BoVW works as follows. From every image in a dataset, relevant areas are identified and a low-level feature vector (called a “descriptor”) is built to represent each area. These vectors, living in what is sometimes called a *descriptor space*, are then grouped into a number of clusters. Each cluster is treated as a discrete visual word, and the clusters will be the *vocabulary* of visual words used to represent all the images in the collection. Now, given a new image, the nearest visual word is identified for each descriptor extracted from it, such that the image can be represented as a BoVW feature vector, by counting the instances of each visual word in the image (note that an occurrence of a low-level descriptor vector in an image, after mapping to the nearest cluster, will increment the count of a single dimension of the higher-level BoVW vector). In our work, the representation of each word (tag) is also a BoVW vector. The values of each dimension are obtained by summing the occurrences of the relevant visual word in all the images tagged with the word. Again, raw counts are transformed into Local Mutual Information scores. The process to extract visual words and use them to create image-based vectors to represent (real) words is illustrated in Figure 1, for a hypothetical example in which there is only one image in the collection labeled with the word *horse*.

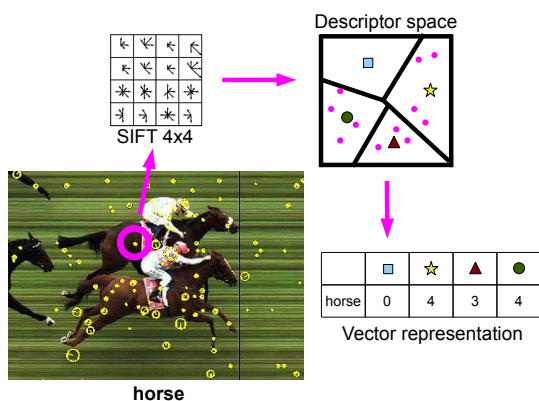


Figure 1: Procedure to build a visual representation for a word, exemplified with SIFT features.

We extract descriptor features of two types.<sup>6</sup> First, the standard Scale-Invariant Feature Transform (**SIFT**) feature vectors (Lowe, 1999; Lowe, 2004), good at characterizing parts of objects. Second, **LAB** features (Fairchild, 2005), which encode only color information. We also experimented with other visual features, such as those focusing on edges (Canny, 1986), texture (Zhu et al., 2002), and shapes (Oliva and Torralba, 2001), but they were not useful for the color tasks. Moreover, we experimented also with different color scales, such as LUV, HSV and RGB, obtaining significantly worse performance compared to LAB. Further details on feature extraction follow.

SIFT features are designed to be invariant to image scale and rotation, and have been shown to provide a robust matching across affine distortion, noise and change in illumination. The version of SIFT features that we use is sensitive to color (RGB scale; LUV, LAB and OPPONENT gave worse results). We automatically identified keypoints for each image and extracted SIFT features on a regular grid defined around the keypoint with five pixels spacing, at four multiple scales (10, 15, 20, 25 pixel radii), zeroing the low contrast ones. To obtain the visual word vocabulary, we cluster the SIFT feature vectors with the standardly used *k*-means clustering algorithm. We varied the number *k* of visual words between 500 and 2,500 in steps of 500.

For the SIFT-based representation of images, we used spatial histograms to introduce weak *geometry* (Grauman and Darrell, 2005; Lazebnik et al., 2006), dividing the image into several (spatial) regions, representing each region in terms of BoVW, and then concatenating the vectors. In our experiments, the spatial regions were obtained by dividing the image in  $4 \times 4$ , for a total of 16 regions (other values and a global representation did not perform as well). Note that, following standard practice, descriptor clustering was performed ignoring the region partition, but the resulting visual words correspond to different dimensions in the concatenated BoVW vectors, depending on the region in which they occur. Consequently, a vocabulary of *k* visual words results in BoVW vectors with  $k \times 16$  dimensions.

<sup>6</sup>We use VLFeat (<http://www.vlfeat.org/>) for feature extraction (Vedaldi and Fulkerson, 2008).

The LAB color space plots image data in 3 dimensions along 3 independent (orthogonal) axes, one for brightness (luminance) and two for color (chrominance). Luminance corresponds closely to brightness as recorded by the brain-eye system; the chrominance (red-green and yellow-blue) axes mimic the oppositional color sensations the retina reports to the brain (Szeliski, 2010). LAB features are densely sampled for each pixel. Also here we use the  $k$ -means algorithm to build the descriptor space. We varied the number of  $k$  visual words between 128 and 1,024 in steps of 128.

### 2.3 Multimodal models

To assemble the textual and visual representations in multimodal semantic spaces, we concatenate the two vectors after normalizing them. We use the linear weighted combination function proposed by Bruni et al. (2011): Given a word that is present both in the textual model and in the visual model, we separately normalize the two vectors  $F_t$  and  $F_v$  and we combine them as follows:

$$F = \alpha \times F_t \oplus (1 - \alpha) \times F_v$$

where  $\oplus$  is the vector concatenate operator. The weighting parameter  $\alpha$  ( $0 \leq \alpha \leq 1$ ) is tuned on the MEN development data (2,000 word pairs; details on the MEN dataset in the next section). We find the optimal value to be close to  $\alpha = 0.5$  for most model combinations, suggesting that textual and visual information should have similar weight. Our implementation of the proposed method is open source and publicly available.<sup>7</sup>

### 2.4 Hybrid models

We further introduce hybrid models that exploit the patterns of co-occurrence of words as tags of the same images. Like textual models, these models are based on word co-occurrence; like visual models, they consider co-occurrence in images (image labels). In one model (**ESP-Win**, analogous to window-based models), words tagging an image were represented in terms of co-occurrence with the other tags in the image label (Baroni and Lenci (2008) are a precedent for the use of ESP-Win). The other (**ESP-Doc**, analogous to document-based

models) represented words in terms of their co-occurrence with images, using each image as a different dimension. This information is very easy to extract, as it does not require the sophisticated techniques used in computer vision. We expected these models to perform very bad; however, as we will show, they perform relatively well in all but one of the tasks tested.

## 3 Textual and visual models as general semantic models

We test the models just presented in two different ways: First, as general models of word meaning, testing their correlation to human judgements on word similarity and relatedness (this section). Second, as models of the meaning of color terms (sections 4 and 5).

We use one standard dataset (**WordSim353**) and one new dataset (**MEN**). WordSim353 (Finkelstein et al., 2002) is a widely used benchmark constructed by asking 16 subjects to rate a set of 353 word pairs on a 10-point similarity scale and averaging the ratings (*dollar/buck* receives a high 9.22 average rating, *professor/cucumber* a low 0.31). MEN is a new evaluation benchmark with a better coverage of our multimodal semantic models.<sup>8</sup> It contains 3,000 pairs of randomly selected words that occur as ESP tags (pairs sampled to ensure a balanced range of relatedness levels according to a text-based semantic score). Each pair is scored on a  $[0, 1]$ -normalized semantic relatedness scale via ratings obtained by crowdsourcing on the Amazon Mechanical Turk (refer to the online MEN documentation for more details). For example, *cold/frost* has a high 0.9 MEN score, *eat/hair* a low 0.1. We evaluate the models in terms of their Spearman correlation to the human ratings. Our models have a perfect MEN coverage and a coverage of 252 WordSim pairs.

We used the development set of MEN to test the effect of varying the number  $k$  of visual words in SIFT and LAB. We restrict the discussion to SIFT with the optimal  $k$  (2.5K words) and to LAB with the optimal (256), lowest (128), and highest  $k$  (1024). We report the results of the multimodal

<sup>8</sup>An updated version of MEN is available from <http://clic.cimec.unitn.it/~elia.bruni/MEN.html>. The version used here contained 10 judgements per word pair.

<sup>7</sup><https://github.com/s2m/FUSE>

models built with these visual models and the best textual models (Window2 and Window20).

Columns WS and MEN in Table 1 report correlations with the WordSim and MEN ratings, respectively. As expected, because they are more mature and capture a broader range of semantic information, textual models perform much better than purely visual models. Also as expected, SIFT features outperform the simpler LAB features for this task.

A first indication that visual information helps is the fact that, for MEN, multimodal models perform best. Note that all models that are sensitive to visual information perform better for MEN than for WordSim, and the reverse is true for textual models. Because of its design, word pairs in MEN can be expected to be more imageable than those in WordSim, so the visual information is more relevant for this dataset. Also recall that we did some parameter tuning on held-out MEN data.

Surprisingly, hybrid models perform quite well: They are around 10 points worse than textual and multimodal models for WordSim, and only slightly worse than multimodal models for MEN.

## 4 Experiment 1: Discovering the color of concrete objects

In Experiment 1, we test the hypothesis that the relation between words denoting concrete things and words denoting their typical color is reflected by the distance of the corresponding vectors better when the models are sensitive to visual information.

### 4.1 Method

Two authors labeled by consensus a list of concrete nouns (extracted from the BLESS dataset<sup>9</sup> and the nouns in the BNC occurring with color terms more than 100 times) with one of the 11 colors from the basic set proposed by Berlin and Kay (1969): *black, blue, brown, green, grey, orange, pink, purple, red, white, yellow*. Objects that do not have an obvious characteristic color (*computer*) and those with more than one characteristic color (*zebra, bear*) were eliminated. Moreover, only nouns covered by all the models were preserved. The final list con-

<sup>9</sup><http://sites.google.com/site/geometricalmodels/shared-evaluation>

<i>Model</i>	<i>WS</i>	<i>MEN</i>	<i>E1</i>	<i>E2</i>
DM	.44	.42	3 (09)	.14
Document	.63	.62	3 (07)	.06
Window2	<b>.70</b>	.66	5 (13)	.49***
Window20	<b>.70</b>	.62	3 (11)	.53***
LAB <sub>128</sub>	.21	.41	<b>1</b> (27)	.25*
LAB <sub>256</sub>	.21	.41	2 (24)	.24*
LAB <sub>1024</sub>	.19	.41	2 (24)	.28**
SIFT <sub>2.5K</sub>	.33	.44	3 (15)	.57***
W2-LAB <sub>128</sub>	.40	.59	<b>1</b> (27)	.40***
W2-LAB <sub>256</sub>	.41	.60	2 (23)	.40***
W2-LAB <sub>1024</sub>	.39	.61	2 (24)	.44***
W20-LAB <sub>128</sub>	.40	.60	<b>1</b> (27)	.36***
W20-LAB <sub>256</sub>	.41	.60	2 (23)	.36***
W20-LAB <sub>1024</sub>	.39	.62	2 (24)	.40***
W2-SIFT <sub>2.5K</sub>	.64	<b>.69</b>	2.5 (19)	.68***
W20-SIFT <sub>2.5K</sub>	.64	.68	2 (17)	<b>.73</b> ***
ESP-Doc	.52	.66	<b>1</b> (37)	.29*
ESP-Win	.55	.68	4 (15)	.16

Table 1: Results of the textual, visual, multimodal, and hybrid models on the general semantic tasks (first two columns, section 3; Pearson  $\rho$ ) and Experiments 1 (E1, section 4) and 2 (E2, section 5). E1 reports the median rank of the correct color and the number of top matches (in parentheses), and E2 the average difference in normalized cosines between literal and nonliteral adjective-noun phrases, with the significance of a t-test (\*\*\* for  $p < 0.001$ , \*\*  $< 0.01$ , \*  $< 0.05$ ).

tains 52 nouns.<sup>10</sup> Some random examples are *fog–grey, crow–black, wood–brown, parsley–green, and grass–green*.

For evaluation, we measured the cosine of each noun with the 11 basic color words in the space produced by each model, and recorded the rank of the correct color in the resulting ordered list.

### 4.2 Results

Column E1 in Table 1 reports the median rank for each model (the smaller the rank, the better the model), as well as the number of exact matches (that is, number of nouns for which the model ranks the correct color first).

Discovering knowledge such that grass is green is arguably a simple task but Experiment 1 shows

<sup>10</sup>Dataset available from the second author’s webpage, under resources.



that textual models fail this simple task, with median ranks around 3.<sup>11</sup> This is consistent with the findings in Baroni and Lenci (2008) that standard distributional models do not capture the association between concrete concepts and their typical attributes. Visual models, as expected, are better at capturing the association between concepts and visual attributes. In fact, all models that are sensitive to visual information achieve median rank 1.

Multimodal models do not increase performance with respect to visual models: For instance, both W2-LAB<sub>128</sub> and W20-LAB<sub>128</sub> have the same median rank and number of exact matches as LAB<sub>128</sub> alone. Textual information in this case is not complementary to visual information, but simply poorer.

Also note that LAB features do better than SIFT features. This is probably due to the fact that Experiment 1 is basically about identifying a large patch of color. The SIFT features we are using are also sensitive to color, but they seem to be misguided by the other cues that they extract from images. For example, pigs are pink in LAB space but brown in SIFT space, perhaps because SIFT focused on the color of the typical environment of a pig. We can thus confirm that, by limiting multimodal spaces to SIFT features, as has been done until now in the literature, we are missing important semantic information, such as the color information that we can mine with LAB.

Again we find that hybrid models do very well, in fact in this case they have the top performance, as they perform better than LAB<sub>128</sub> (the difference, which can be noticed in the number of exact matches, is highly significant according to a paired Mann-Whitney test, with  $p < 0.001$ ).

## 5 Experiment 2

Experiment 2 requires more sophisticated information than Experiment 1, as it involves distinguishing between literal and nonliteral uses of color terms.

---

<sup>11</sup>We also experimented with a model based on direct co-occurrence of adjectives and nouns, obtaining promising results in a preliminary version of Exp. 1. We abandoned this approach because such a model inherently lacks scalability, as it will not generalize behind cases where the training data contain direct examples of co-occurrences of the target pairs.

## 5.1 Method

We test the performance of the different models with a dataset consisting of color adjective-noun phrases, randomly drawn from the most frequent 8K nouns and 4K adjectives in the concatenated ukWaC, Wackypedia, and BNC corpora (four color terms are not among these, so the dataset includes phrases for *black*, *blue*, *brown*, *green*, *red*, *white*, and *yellow* only). These were tagged by consensus by two human judges as **literal** (*white towel*, *black feather*) or **nonliteral** (*white wine*, *white musician*, *green future*). Some phrases had both literal and nonliteral uses, such as *blue book* in “book that is blue” vs. “automobile price guide”. In these cases, only the most common sense (according to the judges) was taken into account for the present experiment. The dataset consists of 370 phrases, of which our models cover 342, 227 literal and 115 nonliteral.<sup>12</sup>

The prediction is that, in good semantic models, literal uses will in general result in a higher similarity between the noun and color term vectors: A white towel is white, while wine or musicians are not white in the same manner. We test this prediction by comparing the average cosine between the color term and the nouns across the literal and nonliteral pairs (similar results were obtained in an evaluation in terms of prediction accuracy of a simple classifier).

## 5.2 Results

Column E2 in Table 1 summarizes the results of the experiment, reporting the mean difference between the normalized cosines (that is, how large the difference is between the literal and nonliteral uses of color terms), as well as the significance of the differences according to a t-test. Window-based models perform best among textual models, particularly Window20, while the rest can’t discriminate between the two uses. This is particularly striking for the Document model, which performs quite well in general semantic tasks but bad in visual tasks.

Visual models are all able to discriminate between the two uses, suggesting that indeed visual information can capture nonliteral aspects of meaning. However, in this case SIFT features perform much better than LAB features, as Experiment 2 involves

---

<sup>12</sup>Dataset available upon request to the second author.

tackling much more sophisticated information than Experiment 1. This is consistent with the fact that, for LAB, a lower  $k$  (lower granularity of the information) performs better for Experiment 1 and a higher  $k$  (higher granularity) for Experiment 2.

One crucial question to ask, given the goals of our research, is whether textual and visual models are doing essentially the same job, only using different types of information. Note that, in this case, multimodal models increase performance over the individual modalities, and are the best models for this task. This suggests that the information used in the individual models is complementary, and indeed there is no correlation between the cosines obtained with the best textual and visual models (Pearson’s  $\rho = .09$ ,  $p = .11$ ).

Figure 2 depicts the results broken down by color.<sup>13</sup> Both modalities can capture the differences for *black* and *green*, probably because nonliteral uses of these color terms have also clear textual correlates (more concretely, topical correlates, as they are related to race and ecology, respectively).<sup>14</sup> Significantly, however, vision can capture nonliteral uses of *blue* and *red*, while text can’t. Note that these uses (*blue note*, *shark*, *shield*, *red meat*, *district*, *face*) do not have a clear topical correlate, and thus it makes sense that vision does a better job.

Finally, note that for this more sophisticated task, hybrid models perform quite bad, which shows their limitations as models of word meaning.<sup>15</sup> Overall,

<sup>13</sup>*Yellow* and *brown* are excluded because the dataset contains only one and two instances of nonliteral cases for these terms, respectively. The significance of the differences as explained in the text has been tested via t-tests.

<sup>14</sup>It’s not entirely clear why neither modality can capture the differences for *white*; for text, it may be because the nonliteral cases are not so tied to race as is the cases for *black*, but they also contain many other types of nonliteral uses, such as type-referring (*white wine/rice/cell*) or metonymical ones (*white smile*).

<sup>15</sup>The hybrid model that performs best in the color tasks is ESP-Doc. This model can only detect a relation between an adjective and a noun if they directly co-occur in the label of at least one image (a “document” in this setting). The more direct co-occurrences there are, the more related the words will be for the model. This works for Exp. 1: Since the ESP labels are lists of what subjects saw in a picture, and the adjectives of Exp. 1 are typical colors of objects, there is a high co-occurrence, as all but one adjective-noun pairs co-occur in at least one ESP label. For the model to perform well in Exp. 2 too, literal phrases should occur in the same labels and non-literal pairs should not. We

our results suggest that co-occurrence in an image label can be used as a surrogate of true visual information to some extent, but the behavior of hybrid models depends on ad-hoc aspects of the labeled dataset, and, from an empirical perspective, they are more limited than truly multimodal models, because they require large amounts of rich verbal picture descriptions to reach good coverage.

## 6 Related work

There is an increasing amount of work in computer vision that exploits text-derived information for image retrieval and annotation tasks (Farhadi et al., 2010; Kulkarni et al., 2011). One particular technique inspired by NLP that has acted as a very effective proxy from CV to NLP is precisely the BoVW. Recently, NLPers have begun exploiting BoVW to enrich distributional models that represent word meaning with visual features automatically extracted from images (Feng and Lapata, 2010; Bruni et al., 2011; Leong and Mihalcea, 2011). Previous work in this area relied on SIFT features only, whereas we have enriched the visual representation of words with other kinds of features from computer vision, namely, color-related features (LAB). Moreover, earlier evaluation of multimodal models has focused only on standard word similarity tasks (using mainly WordSim353), whereas we have tested them on both general semantic tasks and specific tasks that tap directly into aspects of semantics (such as color) where we expect visual information to be crucial.

The most closely related work to ours is that recently presented by Özbal et al. (2011). Like us, Özbal and colleagues use both a textual model and a visual model (as well as Google adjective-noun co-occurrence counts) to find the typical color of an object. However, their visual model works by analyzing pictures associated with an object, and determining the color of the object directly by image analysis. We attempt the more ambitious goal of separately associating a vector to nouns and adjectives, and de-

find no such difference (89% of adjective-noun pairs co-occur in at least one image in the literal set, 86% in the nonliteral set), because many of the relevant pairs describe concrete concepts that, while not necessarily of the “right” literal colour, are perfectly fit to be depicted in images (“blue shark”, “black boy”, “white wine”).

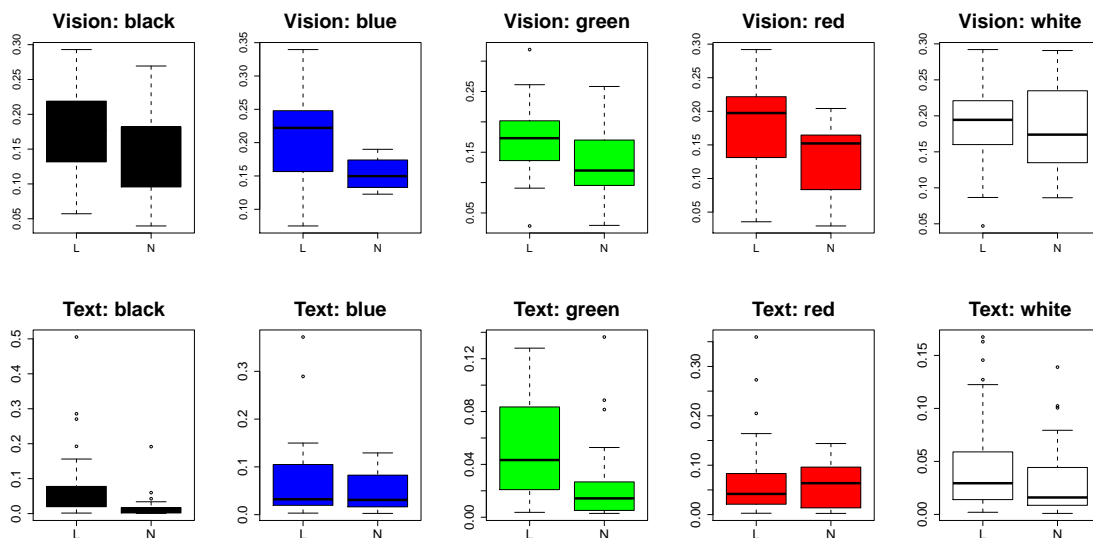


Figure 2: Discrimination of literal (L) vs. nonliteral (N) uses by the best visual and textual models.

termining the color of an object by the nearness of the noun denoting the object to the color term. In other words, we are trying to model the meaning of color *terms* and how they relate to other words, and not to directly extract the color of an object from pictures depicting them. Our second experiment is connected to the literature on the automated detection of figurative language (Shutova, 2010). There is in particular some similarity with the tasks studied by Turney et al. (2011). Turney and colleagues try, among other things, to distinguish literal and metaphorical usages of adjectives when combined with nouns, including the highly visual adjective *dark* (*dark hair* vs. *dark humour*). Their method, based on automatically quantifying the degree of abstractness of the noun, is complementary to ours. Future work could combine our approach and theirs.

## 7 Conclusion

We have presented evidence that distributional semantic models based on text, while providing a good general semantic representation of word meaning, can be outperformed by models using visual information for semantic aspects of words where vision is relevant. More generally, this suggests that computer vision is mature enough to significantly contribute to perceptually grounded computational models of language. We have also shown

that different types of visual features (LAB, SIFT) are appropriate for different tasks. Future research should investigate automated methods to discover which (if any) kind of visual information should be highlighted in which task, more sophisticated multimodal models, visual properties other than color, and larger color datasets, such as the one recently introduced by Mohammad (2011).

## Acknowledgments

E.B. and M.B. are partially supported by a Google Research Award. G.B. is partially supported by the Spanish Ministry of Science and Innovation (FFI2010-15006, TIN2009-14715-C04-04), the EU PASCAL2 Network of Excellence (FP7-ICT-216886) and the AGAUR (2010 BP-A 00070). The E2 evaluation set was created by G.B. with Louise McNally and Eva Maria Vecchi. Fig. 1 was adapted from a figure by Jasper Uijlings. G. B. thanks Margarita Torrent for taking care of her children while she worked hard to meet the Sunday deadline.

## References

- Marco Baroni and Alessandro Lenci. 2008. Concepts and properties in word spaces. *Italian Journal of Linguistics*, 20(1):55–88.
- Marco Baroni and Alessandro Lenci. 2010. Distributional Memory: A general framework for

- corpus-based semantics. *Computational Linguistics*, 36(4):673–721.
- Shane Bergsma and Randy Goebel. 2011. Using visual information to predict lexical preference. In *Proceedings of Recent Advances in Natural Language Processing*, pages 399–405, Hissar.
- Shane Bergsma and Benjamin Van Durme. 2011. Learning bilingual lexicons using the visual similarity of labeled web images. In *Proc. IJCAI*, pages 1764–1769, Barcelona, Spain, July.
- Brent Berlin and Paul Key. 1969. *Basic Color Terms: Their Universality and Evolution*. University of California Press, Berkeley, CA.
- Anna Bosch, Andrew Zisserman, and Xavier Munoz. 2007. Image Classification using Random Forests and Ferns. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8.
- Elia Bruni, Giang Binh Tran, and Marco Baroni. 2011. Distributional semantics from text and images. In *Proceedings of the EMNLP GEMS Workshop*, pages 22–32, Edinburgh.
- John Canny. 1986. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(4):679–698.
- Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. 2004. Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22.
- Stefan Evert. 2005. *The Statistics of Word Cooccurrences*. Dissertation, Stuttgart University.
- Mark D. Fairchild. 2005. Status of cie color appearance models.
- A. Farhadi, M. Hejrati, M. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. 2010. Every picture tells a story: Generating sentences from images. In *Proceedings of ECCV*.
- Yansong Feng and Mirella Lapata. 2010. Visual information in semantic representation. In *Proceedings of HLT-NAACL*, pages 91–99, Los Angeles, CA.
- Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppín. 2002. Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.
- Kristen Grauman and Trevor Darrell. 2005. The pyramid match kernel: Discriminative classification with sets of image features. In *In ICCV*, pages 1458–1465.
- G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. Berg, and T. Berg. 2011. Baby talk: Understanding and generating simple image descriptions. In *Proceedings of CVPR*.
- Thomas Landauer and Susan Dumais. 1997. A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.
- Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2, CVPR 2006*, pages 2169–2178, Washington, DC, USA. IEEE Computer Society.
- Chee Wee Leong and Rada Mihalcea. 2011. Going beyond text: A hybrid image-text approach for measuring word relatedness. In *Proceedings of IJCNLP*, pages 1403–1407, Chiang Mai, Thailand.
- Max Louwerse. 2011. Symbol interdependency in symbolic and embodied cognition. *Topics in Cognitive Science*, 3:273–302.
- David Lowe. 1999. Object Recognition from Local Scale-Invariant Features. *Computer Vision, IEEE International Conference on*, 2:1150–1157 vol.2, August.
- David Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), November.
- Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods*, 28:203–208.
- Saif Mohammad. 2011. Colourful language: Measuring word-colour associations. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 97–106, Portland, Oregon.
- Raymond J. Mooney. 2008. Learning to connect language and perception.
- David Nister and Henrik Stewenius. 2006. Scalable recognition with a vocabulary tree. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2, CVPR ’06*, pages 2161–2168.
- Aude Oliva and Antonio Torralba. 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. *Int. J. Comput. Vision*, 42:145–175.
- Gözde Özbal, Carlo Strapparava, Rada Mihalcea, and Daniele Pighin. 2011. A comparison of unsupervised methods to associate colors with words. In *Proceedings of ACL*, pages 42–51, Memphis, TN.
- Ekaterina Shutova. 2010. Models of metaphor in NLP. In *Proceedings of ACL*, pages 688–697, Uppsala, Sweden.
- Josef Sivic and Andrew Zisserman. 2003. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1470–1477, October.

- Richard Szeliski. 2010. *Computer Vision : Algorithms and Applications*. Springer-Verlag New York Inc.
- Peter Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Peter Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of EMNLP*, pages 680–690, Edinburgh, UK.
- Andrea Vedaldi and Brian Fulkerson. 2008. VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>.
- Luis von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 319–326, Vienna, Austria.
- Jun Yang, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. 2007. Evaluating bag-of-visual-words representations in scene classification. In *Multimedia Information Retrieval*, pages 197–206.
- Song Chun Zhu, Cheng en Guo, Ying Nian Wu, and Yizhou Wang. 2002. What are textons? In *Computer Vision - ECCV 2002, 7th European Conference on Computer Vision, Copenhagen, Denmark, May 28-31, 2002, Proceedings, Part IV*, pages 793–807. Springer.

# A Class-Based Agreement Model for Generating Accurately Inflected Translations

Spence Green

Computer Science Department, Stanford University  
spenceg@stanford.edu

John DeNero

Google  
denero@google.com

## Abstract

When automatically translating from a weakly inflected source language like English to a target language with richer grammatical features such as gender and dual number, the output commonly contains morpho-syntactic agreement errors. To address this issue, we present a target-side, class-based agreement model. Agreement is promoted by scoring a sequence of fine-grained morpho-syntactic classes that are predicted during decoding for each translation hypothesis. For English-to-Arabic translation, our model yields a +1.04 BLEU average improvement over a state-of-the-art baseline. The model does not require bitext or phrase table annotations and can be easily implemented as a feature in many phrase-based decoders.

## 1 Introduction

Languages vary in the degree to which surface forms reflect grammatical relations. English is a weakly inflected language: it has a narrow verbal paradigm, restricted nominal inflection (plurals), and only the vestiges of a case system. Consequently, translation *into* English—which accounts for much of the machine translation (MT) literature (Lopez, 2008)—often involves some amount of morpho-syntactic dimensionality reduction. Less attention has been paid to what happens during translation *from* English: richer grammatical features such as gender, dual number, and overt case are effectively latent variables that must be inferred during decoding. Consider the output of Google Translate for the simple English sentence in Fig. 1. The correct translation is a monotone mapping of the input. However, in Arabic, SVO word order requires both gender and number agreement between the subject *السيارة* ‘the car’ and verb *يذهب* ‘go’. The MT system selects the correct verb stem, but with masculine inflection. Although the translation has

(1) *السيارة يذهب بسرعة*  
the-car<sub>SG.DEF.FEM</sub> go<sub>SG.MASC</sub> with-speed<sub>SG.FEM</sub>  
*The car goes quickly*

Figure 1: Ungrammatical Arabic output of Google Translate for the English input *The car goes quickly*. The subject should agree with the verb in both gender and number, but the verb has masculine inflection. For clarity, the Arabic tokens are arranged left-to-right.

the correct semantics, it is ultimately ungrammatical. This paper addresses the problem of generating text that conforms to morpho-syntactic agreement rules.

Agreement relations that cross statistical phrase boundaries are not explicitly modeled in most phrase-based MT systems (Avramidis and Koehn, 2008). We address this shortcoming with an agreement model that scores sequences of fine-grained morpho-syntactic *classes*. First, bound morphemes in translation hypotheses are segmented. Next, the segments are labeled with classes that encode both syntactic category information (i.e., parts of speech) and grammatical features such as number and gender. Finally, agreement is promoted by scoring the predicted class sequences with a generative Markov model.

Our model scores hypotheses *during decoding*. Unlike previous models for scoring syntactic relations, our model does not require bitext annotations, phrase table features, or decoder modifications. The model can be implemented using the feature APIs of popular phrase-based decoders such as Moses (Koehn et al., 2007) and Phrasal (Cer et al., 2010).

Intuition might suggest that the standard *n*-gram language model (LM) is sufficient to handle agreement phenomena. However, LM statistics are sparse, and they are made sparser by morphological variation. For English-to-Arabic translation, we achieve a +1.04 BLEU average improvement by tiling our model on top of a large LM.

It has also been suggested that this setting requires morphological generation because the bitext may not contain all inflected variants (Minkov et al., 2007; Toutanova et al., 2008; Fraser et al., 2012). However, using lexical coverage experiments, we show that there is ample room for translation quality improvements through better *selection* of forms that already exist in the translation model.

## 2 A Class-based Model of Agreement

### 2.1 Morpho-syntactic Agreement

*Morpho-syntactic agreement* refers to a relationship between two sentence elements *a* and *b* that must have at least one matching grammatical feature.<sup>1</sup> Agreement relations tend to be defined for particular syntactic configurations such as verb-subject, noun-adjective, and pronoun-antecedent. In some languages, agreement affects the surface forms of the words. For example, from the perspective of generative grammatical theory, the lexicon entry for the Arabic nominal السيارة ‘the car’ contains a feminine gender feature. When this nominal appears in the subject argument position, the verb-subject agreement relationship triggers feminine inflection of the verb.

Our model treats agreement as a sequence of scored, pairwise relations between adjacent words. Of course, this assumption excludes some agreement phenomena, but it is sufficient for many common cases. We focus on English-Arabic translation as an example of a translation direction that expresses substantially more morphological information in the target. These relations are best captured in a target-side model because they are mostly unobserved (from lexical clues) in the English source.

The agreement model scores sequences of morpho-syntactic word classes, which express grammatical features relevant to agreement. The model has three components: a segmenter, a tagger, and a scorer.

### 2.2 Morphological Segmentation

Segmentation is a procedure for converting raw surface forms to component morphemes. In some languages, agreement relations exist between *bound morphemes*, which are syntactically independent yet phonologically dependent morphemes. For example,

<sup>1</sup>We use *morpho-syntactic* and *grammatical* agreement interchangeably, as is common in the literature.

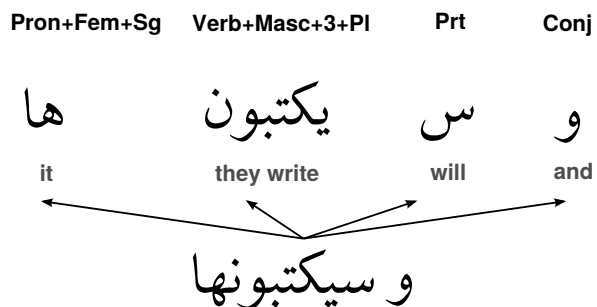


Figure 2: Segmentation and tagging of the Arabic token وسيكتبونها ‘and they will write it’. This token has four segments with conflicting grammatical features. For example, the number feature is singular for the pronominal object and plural for the verb. Our model segments the raw token, tags each segment with a morpho-syntactic class (e.g., “Pron+Fem+Sg”), and then scores the class sequences.

the single raw token in Fig. 2 contains at least four grammatically independent morphemes. Because the morphemes bear conflicting grammatical features and basic parts of speech (POS), we need to segment the token before we can evaluate agreement relations.<sup>2</sup>

Segmentation is typically applied as a bitext pre-processing step, and there is a rich literature on the effect of different segmentation schemata on translation quality (Koehn and Knight, 2003; Habash and Sadat, 2006; El Kholy and Habash, 2012). Unlike previous work, we segment each translation hypothesis as it is generated (i.e., during decoding). This permits greater modeling flexibility. For example, it may be useful to count tokens with bound morphemes as a unit during phrase extraction, but to score segmented morphemes separately for agreement.

We treat segmentation as a character-level sequence modeling problem and train a linear-chain conditional random field (CRF) model (Lafferty et al., 2001). As a pre-processing step, we group contiguous non-native characters (e.g., Latin characters in Arabic text). The model assigns four labels:

- **I**: Continuation of a morpheme
- **O**: Outside morpheme (whitespace)
- **B**: Beginning of a morpheme
- **F**: Non-native character(s)

<sup>2</sup>Segmentation also improves translation of compounding languages such as German (Dyer, 2009) and Finnish (Macherey et al., 2011).

Translation Model	
$e$	Target sequence of $I$ words
$f$	Source sequence of $J$ words
$a$	Sequence of $K$ phrase alignments for $\langle e, f \rangle$
$\Pi$	Permutation of the alignments for target word order $e$
$h$	Sequence of $M$ feature functions
$\lambda$	Sequence of learned weights for the $M$ features
$H$	A priority queue of hypotheses
Class-based Agreement Model	
$t \in T$	Set of morpho-syntactic classes
$s \in S$	Set of all word segments
$\theta_{seg}$	Learned weights for the CRF-based segmenter
$\theta_{tag}$	Learned weights for the CRF-based tagger
$\phi_o, \phi_t$	CRF potential functions (emission and transition)
$\tau$	Sequence of $I$ target-side predicted classes
$\pi$	$T$ dimensional (log) prior distribution over classes
$\hat{s}$	Sequence of $I$ word segments
$\sigma$	Model state: a tagged segment $\langle s, t \rangle$

Figure 3: Notation used in this paper. The convention  $e_i^I$  indicates a subsequence of a length  $I$  sequence.

The features are indicators for (character, position, label) triples for a five character window and bigram label transition indicators.

This formulation is inspired by the classic “IOB” text chunking model (Ramshaw and Marcus, 1995), which has been previously applied to Chinese segmentation (Peng et al., 2004). It can be learned from gold-segmented data, generally applies to languages with bound morphemes, and does not require a hand-compiled lexicon.<sup>3</sup> Moreover, it has only four labels, so Viterbi decoding is very fast. We learn the parameters  $\theta_{seg}$  using a quasi-Newton (QN) procedure with  $l_1$  (lasso) regularization (Andrew and Gao, 2007).

### 2.3 Morpho-syntactic Tagging

After segmentation, we tag each segment with a fine-grained morpho-syntactic class. For this task we also train a standard CRF model on full sentences with gold classes and segmentation. We use the same QN procedure as before to obtain  $\theta_{tag}$ .

A translation derivation is a tuple  $\langle e, f, a \rangle$  where  $e$  is the target,  $f$  is the source, and  $a$  is an alignment between the two. The CRF tagging model predicts a target-side class sequence  $\tau^*$

$$\tau^* = \arg \max_{\tau} \sum_{i=1}^I \theta_{tag} \cdot \{ \phi_o(\tau_i, i, e) + \phi_t(\tau_i, \tau_{i-1}) \}$$

where further notation is defined in Fig. 3.

<sup>3</sup>Mada, the standard tool for Arabic segmentation (Habash and Rambow, 2005), relies on a manually compiled lexicon.

**Set of Classes** The tagger assigns morpho-syntactic classes, which are coarse POS categories refined with grammatical features such as gender and definiteness. The coarse categories are the universal POS tag set described by Petrov et al. (2012). More than 25 treebanks (in 22 languages) can be automatically mapped to this tag set, which includes “Noun” (nominals), “Verb” (verbs), “Adj” (adjectives), and “ADP” (pre- and post-positions). Many of these treebanks also contain per-token morphological annotations. It is easy to combine the coarse categories with selected grammatical annotations.

For Arabic, we used the coarse POS tags plus definiteness and the so-called phi features (gender, number, and person).<sup>4</sup> For example, السيارة ‘the car’ would be tagged “Noun+Def+Sg+Fem”. We restricted the set of classes to observed combinations in the training data, so the model implicitly disallows incoherent classes like “Verb+Def”.

**Features** The tagging CRF includes emission features  $\phi_o$  that indicate a class  $\tau_i$  appearing with various orthographic characteristics of the word sequence being tagged. In typical CRF inference, the entire observation sequence is available throughout inference, so these features can be scored on observed words in an arbitrary neighborhood around the current position  $i$ . However, we conduct CRF inference in tandem with the translation decoding procedure (§3), creating an environment in which subsequent words of the observation are not available; the MT system has yet to generate the rest of the translation when the tagging features for a position are scored. Therefore, we only define emission features on the observed words at the current and previous positions of a class:  $\phi_o(\tau_i, e_i, e_{i-1})$ .

The emission features are word types, prefixes and suffixes of up to three characters, and indicators for digits and punctuation. None of these features are language specific.

Bigram transition features  $\phi_t$  encode local agreement relations. For example, the model learns that the Arabic class “Noun+Fem” is followed by “Adj+Fem” and not “Adj+Masc” (noun-adjective gender agreement).

<sup>4</sup>Case is also relevant to agreement in Arabic, but it is mostly indicated by diacritics, which are absent in unvocalized text.



## 2.4 Word Class Sequence Scoring

The CRF tagger model defines a conditional distribution  $p(\tau|e; \theta_{tag})$  for a class sequence  $\tau$  given a sentence  $e$  and model parameters  $\theta_{tag}$ . That is, the sample space is over class—not word—sequences. However, in MT, we seek a measure of sentence quality  $q(e)$  that is comparable *across* different hypotheses on the beam (much like the  $n$ -gram language model score). Discriminative model scores have been used as MT features (Galley and Manning, 2009), but we obtained better results by scoring the 1-best class sequences with a generative model. We trained a simple add-1 smoothed bigram language model over gold class sequences in the same treebank training data:

$$q(e) = p(\tau) = \prod_{i=1}^I p(\tau_i | \tau_{i-1})$$

We chose a bigram model due to the aggressive recombination strategy in our phrase-based decoder. For contexts in which the LM is guaranteed to back off (for instance, after an unseen bigram), our decoder maintains only the minimal state needed (perhaps only a single word). In less restrictive decoders, higher order scoring models could be used to score longer-distance agreement relations.

We integrate the segmentation, tagging, and scoring models into a self-contained component in the translation decoder.

## 3 Inference during Translation Decoding

Scoring the agreement model as part of translation decoding requires a novel inference procedure. Crucially, the inference procedure does not measurably affect total MT decoding time.

### 3.1 Phrase-based Translation Decoding

We consider the standard phrase-based approach to MT (Och and Ney, 2004). The distribution  $p(e|f)$  is modeled directly using a log-linear model, yielding the following decision rule:

$$e^* = \arg \max_{e, a, \Pi} \left\{ \sum_{m=1}^M \lambda_m h_m(e, f, a, \Pi) \right\} \quad (1)$$

This decoding problem is NP-hard, thus a beam search is often used (Fig. 4). The beam search relies on three operations, two of which affect the agreement model:

**Input:** implicitly defined search space  
generate initial hypotheses and add to  $H$   
set  $H_{final}$  to  $\emptyset$   
while  $H$  is not empty:  
  set  $H_{ext}$  to  $\emptyset$   
  for each hypothesis  $\eta$  in  $H$ :  
    if  $\eta$  is a goal hypothesis:  
      add  $\eta$  to  $H_{final}$   
    else Extend  $\eta$  and add to  $H_{ext}$      ▶ Score agreement  
  Recombine and Prune  $H_{ext}$   
  set  $H$  to  $H_{ext}$   
**Output:** argmax of  $H_{final}$

Figure 4: Breadth-first beam search algorithm of Och and Ney (2004). Typically, a hypothesis stack  $H$  is maintained for each unique source coverage set.

**Input:**  $(e_1^I, n, is\_goal)$   
run segmenter on attachment  $e_{n+1}^I$  to get  $\hat{s}_1^L$   
get model state  $\sigma = \langle s, t \rangle$  for translation prefix  $e_1^n$   
initialize  $\pi$  to  $-\infty$   
set  $\pi(t) = 0$   
compute  $\tau^*$  from parameters  $\langle s, \hat{s}_1^L, \pi, is\_goal \rangle$   
compute  $q(e_{n+1}^I) = p(\tau^*)$  under the generative LM  
set model state  $\sigma_{new} = \langle \hat{s}_L, \tau_L^* \rangle$  for prefix  $e_1^I$   
**Output:**  $q(e_{n+1}^I)$

Figure 5: Procedure for scoring agreement for each hypothesis generated during the search algorithm of Fig. 4. In the extended hypothesis  $e_1^I$ , the index  $n + 1$  indicates the start of the new attachment.

- Extend a hypothesis with a new phrase pair
- Recombine hypotheses with identical states

We assume familiarity with these operations, which are described in detail in (Och and Ney, 2004).

### 3.2 Agreement Model Inference

The class-based agreement model is implemented as a feature function  $h_m$  in Eq. (1). Specifically, when Extend generates a new hypothesis, we run the algorithm shown in Fig. 5. The inputs are a translation hypothesis  $e_1^I$ , an index  $n$  distinguishing the *prefix* from the *attachment*, and a flag indicating if their concatenation is a goal hypothesis.

The beam search maintains state for each derivation, the score of which is a linear combination of the feature values. States in this program depend on some amount of lexical history. With a trigram language model, the state might be the last two words of the translation prefix. Recombine can be applied to any two hypotheses with equivalent states. As a

result, two hypotheses with different full prefixes—and thus potentially different sequences of agreement relations—can be recombined.

**Incremental Greedy Decoding** Decoding with the CRF-based tagger model in this setting requires some slight modifications to the Viterbi algorithm. We make a greedy approximation that permits recombination and works well in practice. The agreement model state is the *last tagged segment*  $\langle s, t \rangle$  of the concatenated hypothesis. We tag a new attachment by assuming a prior distribution  $\pi$  over the starting position such that  $\pi(t) = 0$  and  $-\infty$  for all other classes, a deterministic distribution in the tropical semiring. This forces the Viterbi path to go through  $t$ . We only tag the final boundary symbol for goal hypotheses.

To accelerate tagger decoding in our experiments, we also used tagging dictionaries for frequently observed word types. For each word type observed more than 100 times in the training data, we restricted the set of possible classes to the set of observed classes.

### 3.3 Translation Model Features

The agreement model score is one decoder feature function. The output of the procedure in Fig. 5 is the log probability of the class sequence of each attachment. Summed over all attachments, this gives the log probability of the whole class sequence.

We also add a new length penalty feature. To discriminate between hypotheses that might have the same number of raw tokens, but different underlying segmentations, we add a penalty equal to the length difference between the segmented and unsegmented attachments  $|\hat{s}_1^L| - |e_{n+1}^L|$ .

## 4 Related Work

We compare our class-based model to previous approaches to scoring syntactic relations in MT.

**Unification-based Formalisms** Agreement rules impose syntactic and semantic constraints on the structure of sentences. A principled way to model these constraints is with a unification-based grammar (UBG). Johnson (2003) presented algorithms for learning and parsing with stochastic UBGs. However, training data for these formalisms remains extremely limited, and it is unclear how to learn such knowledge-rich representations from unlabeled data. One partial

solution is to manually extract unification rules from phrase-structure trees. Williams and Koehn (2011) annotated German trees, and extracted translation rules from them. They then specified manual unification rules, and applied a penalty according to the number of unification failures in a hypothesis. In contrast, our class-based model does not require any manual rules and scores similar agreement phenomena as probabilistic sequences.

**Factored Translation Models** Factored translation models (Koehn and Hoang, 2007) facilitate a more data-oriented approach to agreement modeling. Words are represented as a vector of features such as lemma and POS. The bitext is annotated with separate models, and the annotations are saved during phrase extraction. Hassan et al. (2007) noticed that the target-side POS sequences could be scored, much as we do in this work. They used a target-side LM over Combinatorial Categorical Grammar (CCG) supertags, along with a penalty for the number of operator violations, and also modified the phrase probabilities based on the tags. However, Birch et al. (2007) showed that this approach captures the same re-ordering phenomena as lexicalized re-ordering models, which were not included in the baseline. Birch et al. (2007) then investigated source-side CCG supertag features, but did not show an improvement for Dutch-English.

Subotin (2011) recently extended factored translation models to hierarchical phrase-based translation and developed a discriminative model for predicting target-side morphology in English-Czech. His model benefited from gold morphological annotations on the target-side of the 8M sentence bitext.

In contrast to these methods, our model does not affect phrase extraction and does not require annotated translation rules.

**Class-based LMs** Class-based LMs (Brown et al., 1992) reduce lexical sparsity by placing words in equivalence classes. They have been widely used for speech recognition, but not for MT. Och (1999) showed a method for inducing bilingual word classes that placed each phrase pair into a two-dimensional equivalence class. To our knowledge, Uszkoreit and Brants (2008) are the only recent authors to show an improvement in a state-of-the-art MT system using class-based LMs. They used a classical exchange algorithm for clustering, and learned 512 classes from

a large monolingual corpus. Then they mixed the classes into a word-based LM. However, both Och (1999) and Uszkoreit and Brants (2008) relied on automatically induced classes. It is unclear if their classes captured agreement information.

Monz (2011) recently investigated parameter estimation for POS-based language models, but his classes did not include inflectional features.

**Target-Side Syntactic LMs** Our agreement model is a form of syntactic LM, of which there is a long history of research, especially in speech processing.<sup>5</sup> Syntactic LMs have traditionally been too slow for scoring during MT decoding. One exception was the quadratic-time dependency language model presented by Galley and Manning (2009). They applied a quadratic time dependency parser to every hypothesis during decoding. However, to achieve quadratic running time, they permitted ill-formed trees (e.g., parses with multiple roots). More recently, Schwartz et al. (2011) integrated a right-corner, incremental parser into Moses. They showed a large improvement for Urdu-English, but decoding slowed by three orders of magnitude.<sup>6</sup> In contrast, our class-based model encodes shallow syntactic information without a noticeable effect on decoding time.

Our model can be viewed as a way to score local syntactic relations without extensive decoder modifications. For long-distance relations, Shen et al. (2010) proposed a new decoder that generates target-side dependency trees. The target-side structure enables scoring hypotheses with a trigram dependency LM.

## 5 Experiments

We first evaluate the Arabic segmenter and tagger components independently, then provide English-Arabic translation quality results.

### 5.1 Intrinsic Evaluation of Components

**Experimental Setup** All experiments use the Penn Arabic Treebank (ATB) (Maamouri et al., 2004) parts 1–3 divided into training/dev/test sections according to the canonical split (Rambow et al., 2005).<sup>7</sup>

<sup>5</sup>See (Zhang, 2009) for a comprehensive survey.

<sup>6</sup>In principle, their parser should run in linear time. An implementation issue may account for the decoding slowdown. (*p.c.*)

<sup>7</sup>LDC catalog numbers: LDC2008E61 (ATBp1v4), LDC2008E62 (ATBp2v3), and LDC2008E22 (ATBp3v3.1).

	FULL (%)	INCREMENTAL (%)
Segmenter	98.6	–
Tagger	96.3	96.2

Table 1: Intrinsic evaluation accuracy [%] (development set) for Arabic segmentation and tagging.

The ATB contains clitic-segmented text with *per-segment* morphological analyses (in addition to phrase-structure trees, which we discard). For training the segmenter, we used markers in the vocalized section to construct the IOB character sequences. For training the tagger, we automatically converted the ATB morphological analyses to the fine-grained class set. This procedure resulted in 89 classes.

For the segmentation evaluation, we report *per-character* labeling accuracy.<sup>8</sup> For the tagger, we report *per-token* accuracy.

**Results** Tbl. 1 shows development set accuracy for two settings. FULL is a standard evaluation in which features may be defined over the whole sentence. This includes next-character segmenter features and next-word tagger features. INCREMENTAL emulates the MT setting in which the models are restricted to current and previous observation features. Since the segmenter operates at the character level, we can use the same feature set. However, next-observation features must be removed from the tagger. Nonetheless, tagging accuracy only decreases by 0.1%.

### 5.2 Translation Quality

**Experimental Setup** Our decoder is based on the phrase-based approach to translation (Och and Ney, 2004) and contains various feature functions including phrase relative frequency, word-level alignment statistics, and lexicalized re-ordering models (Tillmann, 2004; Och et al., 2004). We tuned the feature weights on a development set using lattice-based minimum error rate training (MERT) (Macherey et al.,

The data was pre-processed with packages from the Stanford Arabic parser (Green and Manning, 2010). The corpus split is available at <http://nlp.stanford.edu/projects/arabic.shtml>.

<sup>8</sup>We ignore orthographic re-normalization performed by the annotators. For example, they converted the contraction ‘ل’ // back to ‘ل ال’ / Al. As a result, we can report accuracy since the guess and gold segmentations have equal numbers of non-whitespace characters.

	MT04 (tune)		MT02		MT03		MT05		Avg
Baseline	18.14		23.87		18.88		22.60		
+POS	18.11	-0.03	23.65	-0.22	18.99	+0.11	22.29	-0.31	-0.17
+POS+Agr	18.86	<b>+0.72</b>	24.84	<b>+0.97</b>	20.26	<b>+1.38</b>	23.48	<b>+0.88</b>	+1.04
<i>genres</i>	nw		nw		nw		nw		
<i>#sentences</i>	1353		728		663		1056		2447

Table 2: Translation quality results (BLEU-4 [%]) for newswire (nw) sets. Avg is the weighted averaged (by number of sentences) of the individual test set gains. All improvements are statistically significant at  $p \leq 0.01$ .

	MT06		MT08		Avg
Baseline	14.68		14.30		
+POS	14.57	-0.11	14.30	+0.0	-0.06
+POS+Agr	15.04	<b>+0.36</b>	14.49	<b>+0.19</b>	+0.29
<i>genres</i>	nw,bn,ng		nw,ng,wb		
<i>#sentences</i>	1797		1360		3157

Table 3: Mixed genre test set results (BLEU-4 [%]). The MT06 result is statistically significant at  $p \leq 0.01$ ; MT08 is significant at  $p \leq 0.02$ . The genres are: nw, broadcast news (bn), newsgroups (ng), and weblog (wb).

2008). For each set of results, we initialized MERT with uniform feature weights.

We trained the translation model on 502 million words of parallel text collected from a variety of sources, including the Web. Word alignments were induced using a hidden Markov model based alignment model (Vogel et al., 1996) initialized with bilinear parameters from IBM Model 1 (Brown et al., 1993). Both alignment models were trained using two iterations of the expectation maximization algorithm. Our distributed 4-gram language model was trained on 600 million words of Arabic text, also collected from many sources including the Web (Brants et al., 2007).

For development and evaluation, we used the NIST Arabic-English data sets, each of which contains one set of Arabic sentences and multiple English references. To reverse the translation direction for each data set, we chose the first English reference as the source and the Arabic as the reference.

The NIST sets come in two varieties: newswire (MT02-05) and mixed genre (MT06,08). Newswire contains primarily Modern Standard Arabic (MSA), while the mixed genre data sets also contain transcribed speech and web text. Since the ATB contains MSA, and significant lexical and syntactic differences

may exist between MSA and the mixed genres, we achieved best results by tuning on MT04, the largest newswire set.

We evaluated translation quality with BLEU-4 (Papineni et al., 2002) and computed statistical significance with the approximate randomization method of Riezler and Maxwell (2005).<sup>9</sup>

## 6 Discussion of Translation Results

Tbl. 2 shows translation quality results on newswire, while Tbl. 3 contains results for mixed genres. The baseline is our standard system feature set. For comparison, +POS indicates our class-based model trained on the 11 coarse POS tags only (e.g., “Noun”). Finally, +POS+Agr shows the class-based model with the fine-grained classes (e.g., “Noun+Fem+Sg”).

The best result—a +1.04 BLEU average gain—was achieved when the class-based model training data, MT tuning set, and MT evaluation set contained the same genre. We realized smaller, yet statistically significant, gains on the mixed genre data sets. We tried tuning on both MT06 and MT08, but obtained insignificant gains. In the next section, we investigate this issue further.

**Tuning with a Treebank-Trained Feature** The class-based model is trained on the ATB, which is predominantly MSA text. This data set is syntactically regular, meaning that it does not have highly dialectal content, foreign scripts, disfluencies, etc. Conversely, the mixed genre data sets contain more irregularities. For example, 57.4% of MT06 comes from non-newswire genres. Of the 764 newsgroup sentences, 112 contain some Latin script tokens, while others contain very little morphology:

<sup>9</sup>With the implementation of Clark et al. (2011), available at: <http://github.com/jhclark/multeval>.

- (2) تفاح خل كوب 1/2 اخلطي  
 mix 1/2 cup vinegar apple  
*Mix 1/2 cup apple vinegar*
- (3) ماتش ميوزك برنامج بدأ MusicMatch  
 start program miozik maatsh MusicMatch  
*Start the program music match (MusicMatch)*

In these imperatives, there are no lexically marked agreement relations to score. Ex. (2) is an excerpt from a recipe that appears in full in MT06. Ex. (3) is part of usage instructions for the MusicMatch software. The ATB contains few examples like these, so our class-based model probably does not effectively discriminate between alternative hypotheses for these types of sentences.

**Phrase Table Coverage** In a standard phrase-based system, effective translation into a highly inflected target language requires that the phrase table contain the inflected word forms necessary to construct an output with correct agreement. If the requisite words are not present in the search space of the decoder, then no feature function would be sufficient to enforce morpho-syntactic agreement.

During development, we observed that the phrase table of our large-scale English-Arabic system did often contain the inflected forms that we desired the system to select. In fact, correctly agreeing alternatives often appeared in  $n$ -best translation lists. To verify this observation, we computed the lexical coverage of the MT05 reference sentences in the decoder search space. The statistics below report the token-level recall of reference unigrams:<sup>10</sup>

- Baseline system translation output: 44.6%
- Phrase pairs matching source  $n$ -grams: 67.8%

The bottom category includes all lexical items that the decoder could produce in a translation of the source. This large gap between the unigram recall of the actual translation output (top) and the lexical coverage of the phrase-based model (bottom) indicates that translation performance can be improved dramatically by altering the translation model through features such as ours, without expanding the search space of the decoder.

<sup>10</sup>To focus on possibly inflected word forms, we excluded numbers and punctuation from this analysis.

**Human Evaluation** We also manually evaluated the MT05 output for improvements in agreement.<sup>11</sup> Our system produced different output from the baseline for 785 (74.3%) sentences. We randomly sampled 100 of these sentences and counted agreement errors of all types. The baseline contained 78 errors, while our system produced 66 errors, a statistically significant 15.4% error reduction at  $p \leq 0.01$  according to a paired  $t$ -test.

In our output, a frequent source of remaining errors was the case of so-called “deflected agreement”: inanimate plural nouns require feminine singular agreement with modifiers. On the other hand, animate plural nouns require the sound plural, which is indicated by an appropriate masculine or feminine suffix. For example, the inanimate plural *الولايات* ‘states’ requires the singular feminine adjective *المتحدة* ‘united’, not the sound plural *المتحدات*. The ATB does not contain animacy annotations, so our agreement model cannot discriminate between these two cases. However, Alkuhlani and Habash (2011) have recently started annotating the ATB for animacy, and our model could benefit as more data is released.

## 7 Conclusion and Outlook

Our class-based agreement model improves translation quality by promoting local agreement, but with a minimal increase in decoding time and no additional storage requirements for the phrase table. The model can be implemented with a standard CRF package, trained on existing treebanks for many languages, and integrated easily with many MT feature APIs. We achieved best results when the model training data, MT tuning set, and MT evaluation set contained roughly the same genre. Nevertheless, we also showed an improvement, albeit less significant, on mixed genre evaluation sets.

In principle, our class-based model should be more robust to unseen word types and other phenomena that make non-newswire genres challenging. However, our analysis has shown that for Arabic, these genres typically contain more Latin script and transliterated words, and thus there is less morphology to score. One potential avenue of future work would be to adapt our component models to new genres by self-training them on the target side of a large bitext.

<sup>11</sup>The annotator was the first author.

**Acknowledgments** We thank Zhifei Li and Chris Manning for helpful discussions, and Klaus Macherey, Wolfgang Macherey, Daisy Stanton, and Richard Zens for engineering support. This work was conducted while the first author was an intern at Google. At Stanford, the first author is supported by a National Science Foundation Graduate Research Fellowship.

## References

- S. Alkuhlani and N. Habash. 2011. A corpus for modeling morpho-syntactic agreement in Arabic: Gender, number and rationality. In *ACL-HLT*.
- G. Andrew and J. Gao. 2007. Scalable training of  $l_1$ -regularized log-linear models. In *ICML*.
- E. Avramidis and P. Koehn. 2008. Enriching morphologically poor languages for statistical machine translation. In *ACL*.
- A. Birch, M. Osborne, and P. Koehn. 2007. CCG supertags in factored statistical machine translation. In *WMT*.
- T. Brants, A. C. Papat, P. Xu, F. J. Och, and J. Dean. 2007. Large language models in machine translation. In *EMNLP-CoNLL*.
- P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. Della Pietra, and J. C. Lai. 1992. Class-based  $n$ -gram models of natural language. *Computational Linguistics*, 18:467–479.
- 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):263–313.
- D. Cer, M. Galley, D. Jurafsky, and C. D. Manning. 2010. Phrasal: A statistical machine translation toolkit for exploring new model features. In *HLT-NAACL, Demonstration Session*.
- J. H. Clark, C. Dyer, A. Lavie, and N. A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *ACL*.
- C. Dyer. 2009. Using a maximum entropy model to build segmentation lattices for MT. In *NAACL*.
- A. El Kholi and N. Habash. 2012. Orthographic and morphological processing for English-Arabic statistical machine translation. *Machine Translation*, 26(1-2):25–45.
- A. Fraser, M. Weller, A. Cahill, and F. Cap. 2012. Modeling inflection and word-formation in SMT. In *EACL*.
- M. Galley and C. D. Manning. 2009. Quadratic-time dependency parsing for machine translation. In *ACL-IJCNLP*.
- S. Green and C. D. Manning. 2010. Better Arabic parsing: baselines, evaluations, and analysis. In *COLING*.
- 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 *NAACL*.
- H. Hassan, K. Sima'an, and A. Way. 2007. Supertagged phrase-based statistical machine translation. In *ACL*.
- M. Johnson. 2003. Learning and parsing stochastic unification-based grammars. In *COLT*.
- P. Koehn and H. Hoang. 2007. Factored translation models. In *EMNLP-CoNLL*.
- P. Koehn and K. Knight. 2003. Empirical methods for compound splitting. In *EACL*.
- P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL, Demonstration Session*.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.
- A. Lopez. 2008. Statistical machine translation. *ACM Computing Surveys*, 40(8):1–49.
- M. Maamouri, A. Bies, T. Buckwalter, and W. Mekki. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR*.
- W. Macherey, F. Och, I. Thayer, and J. Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *EMNLP*.
- K. Macherey, A. Dai, D. Talbot, A. Papat, and F. Och. 2011. Language-independent compound splitting with morphological operations. In *ACL*.
- E. Minkov, K. Toutanova, and H. Suzuki. 2007. Generating complex morphology for machine translation. In *ACL*.
- C. Monz. 2011. Statistical machine translation with local language models. In *EMNLP*.
- F. J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.
- F. J. Och, D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, et al. 2004. A smorgasbord of features for statistical machine translation. In *HLT-NAACL*.
- F. J. Och. 1999. An efficient method for determining bilingual word classes. In *EACL*.
- K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL*.
- F. Peng, F. Feng, and A. McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *COLING*.
- S. Petrov, D. Das, and R. McDonald. 2012. A universal part-of-speech tagset. In *LREC*.
- O. Rambow, D. Chiang, M. Diab, N. Habash, R. Hwa, et al. 2005. Parsing Arabic dialects. Technical report, Johns Hopkins University.
- L. A. Ramshaw and M. Marcus. 1995. Text chunking using transformation-based learning. In *Proc. of the Third Workshop on Very Large Corpora*.
- S. Riezler and J. T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing in MT. In *ACL-05 Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization (MTSE)*.
- L. Schwartz, C. Callison-Burch, W. Schuler, and S. Wu. 2011. Incremental syntactic language models for phrase-based translation. In *ACL-HLT*.
- L. Shen, J. Xu, and R. Weischedel. 2010. String-to-dependency statistical machine translation. *Computational Linguistics*, 36(4):649–671.

- M. Subotin. 2011. An exponential translation model for target language morphology. In *ACL-HLT*.
- C. Tillmann. 2004. A unigram orientation model for statistical machine translation. In *NAACL*.
- K. Toutanova, H. Suzuki, and A. Ruopp. 2008. Applying morphology generation models to machine translation. In *ACL-HLT*.
- J. Uszkoreit and T. Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *ACL-HLT*.
- S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING*.
- P. Williams and P. Koehn. 2011. Agreement constraints for statistical machine translation into German. In *WMT*.
- Y. Zhang. 2009. *Structured Language Models for Statistical Machine Translation*. Ph.D. thesis, Carnegie Mellon University.

# Deciphering Foreign Language by Combining Language Models and Context Vectors

Malte Nuhn and Arne Mauser\* and Hermann Ney  
Human Language Technology and Pattern Recognition Group  
RWTH Aachen University, Germany  
<surname>@cs.rwth-aachen.de

## Abstract

In this paper we show how to train statistical machine translation systems on real-life tasks using only non-parallel monolingual data from two languages. We present a modification of the method shown in (Ravi and Knight, 2011) that is scalable to vocabulary sizes of several thousand words. On the task shown in (Ravi and Knight, 2011) we obtain better results with only 5% of the computational effort when running our method with an  $n$ -gram language model. The efficiency improvement of our method allows us to run experiments with vocabulary sizes of around 5,000 words, such as a non-parallel version of the VERBMOBIL corpus. We also report results using data from the monolingual French and English GIGAWORD corpora.

## 1 Introduction

It has long been a vision of science fiction writers and scientists to be able to universally communicate in all languages. In these visions, even previously unknown languages can be learned automatically from analyzing foreign language input.

In this work, we attempt to learn statistical translation models from only monolingual data in the source and target language. The reasoning behind this idea is that the elements of languages share statistical similarities that can be automatically identified and matched with other languages.

This work is a big step towards large-scale and large-vocabulary unsupervised training of statistical translation models. Previous approaches have faced constraints in vocabulary or data size. We show how

to scale unsupervised training to real-life translation tasks and how large-scale experiments can be done. Monolingual data is more readily available, if not abundant compared to true parallel or even just translated data. Learning from only monolingual data in real-life translation tasks could improve especially low resource language pairs where few or no parallel texts are available.

In addition to that, this approach offers the opportunity to decipher new or unknown languages and derive translations based solely on the available monolingual data. While we do tackle the full unsupervised learning task for MT, we make some very basic assumptions about the languages we are dealing with:

1. We have large amounts of data available in source and target language. This is not a very strong assumption as books and text on the internet are readily available for almost all languages.
2. We can divide the given text in tokens and sentence-like units. This implies that we know enough about the language to tokenize and sentence-split a given text. Again, for the vast majority of languages, this is not a strong restriction.
3. The writing system is one-dimensional left-to-right. It has been shown (Lin and Knight, 2006) that the writing direction can be determined separately and therefore this assumption does not pose a real restriction.

Previous approaches to unsupervised training for SMT prove feasible only for vocabulary sizes up to around 500 words (Ravi and Knight, 2011) and data

---

\*Author now at Google Inc., amauser@google.com.



sets of roughly 15,000 sentences containing only about 4 tokens per sentence on average. Real data as it occurs in texts such as web pages or news texts does not meet any of these characteristics.

In this work, we will develop, describe, and evaluate methods for large vocabulary unsupervised learning of machine translation models suitable for real-world tasks. The remainder of this paper is structured as follows: In Section 2, we will review the related work and describe how our approach extends existing work. Section 3 describes the model and training criterion used in this work. The implementation and the training of this model is then described in Section 5 and experimentally evaluated in Section 6.

## 2 Related Work

Unsupervised training of statistical translations systems without parallel data and related problems have been addressed before. In this section, we will review previous approaches and highlight similarities and differences to our work. Several steps have been made in this area, such as (Knight and Yamada, 1999), (Ravi and Knight, 2008), or (Snyder et al., 2010), to name just a few. The main difference of our work is, that it allows for much larger vocabulary sizes and more data to be used than previous work while at the same time not being dependent on seed lexica and/or any other knowledge of the languages.

Close to the methods described in this work, Ravi and Knight (2011) treat training and translation without parallel data as a deciphering problem. Their best performing approach uses an EM-Algorithm to train a generative word based translation model. They perform experiments on a Spanish/English task with vocabulary sizes of about 500 words and achieve a performance of around 20 BLEU compared to 70 BLEU obtained by a system that was trained on parallel data. Our work uses the same training criterion and is based on the same generative story. However, we use a new training procedure whose critical parts have constant time and memory complexity with respect to the vocabulary size so that our methods can scale to much larger vocabulary sizes while also being faster.

In a different approach, Koehn and Knight (2002)

induce a bilingual lexicon from only non-parallel data. To achieve this they use a seed lexicon which they systematically extend by using orthographic as well as distributional features such as context, and frequency. They perform their experiments on non-parallel German-English news texts, and test their mappings against a bilingual lexicon. We use a greedy method similar to (Koehn and Knight, 2002) for extending a given lexicon, and we implicitly also use the frequency as a feature. However, we perform fully unsupervised training and do not start with a seed lexicon or use linguistic features.

Similarly, Haghghi et al. (2008) induce a one-to-one translation lexicon only from non-parallel monolingual data. Also starting with a seed lexicon, they use a generative model based on canonical correlation analysis to systematically extend the lexicon using context as well as spelling features. They evaluate their method on a variety of tasks, ranging from inherently parallel data (EUROPARL) to unrelated corpora (100k sentences of the GIGAWORD corpus). They report F-measure scores of the induced entries between 30 to 70. As mentioned above, our work neither uses a seed lexicon nor orthographic features.

## 3 Translation Model

In this section, we describe the statistical training criterion and the translation model that is trained using monolingual data. In addition to the mathematical formulation of the model we describe approximations used.

Throughout this work, we denote the source language words as  $f$  and target language words as  $e$ . The source vocabulary is  $V_f$  and we write the size of this vocabulary as  $|V_f|$ . The same notation holds for the target vocabulary with  $V_e$  and  $|V_e|$ .

As training criterion for the translation model's parameters  $\theta$ , Ravi and Knight (2011) suggest

$$\arg \max_{\theta} \left\{ \prod_f \sum_e P(e) \cdot p_{\theta}(f|e) \right\} \quad (1)$$

We would like to obtain  $\theta$  from Equation 1 using the EM Algorithm (Dempster et al., 1977). This becomes increasingly difficult with more complex translation models. Therefore, we use a simplified

translation model that still contains all basic phenomena of a generic translation process. We formulate the translation process with the same generative story presented in (Ravi and Knight, 2011):

1. Stochastically generate the target sentence according to an  $n$ -gram language model.
2. Insert NULL tokens between any two adjacent positions of the target string with uniform probability.
3. For each target token  $e_i$  (including NULL) choose a foreign translation  $f_i$  (including NULL) with probability  $P_\theta(f_i|e_i)$ .
4. Locally reorder any two adjacent foreign words  $f_{i-1}, f_i$  with probability  $P(\text{SWAP}) = 0.1$ .
5. Remove the remaining NULL tokens.

In practice, however, it is not feasible to deal with the full parameter table  $P_\theta(f_i|e_i)$  which models the lexicon. Instead we only allow translation models where for each *source* word  $f$  the number of words  $e'$  with  $P(f|e') \neq 0$  is below some fixed value. We will refer to this value as the *maximum number of candidates* of the translation model and denote it with  $N_C$ . Note that for a given  $e$  this does not necessarily restrict the number of entries  $P(f'|e) \neq 0$ . Also note that with a fixed value of  $N_C$ , time and memory complexity of the EM step is  $\mathcal{O}(1)$  with respect to  $|V_e|$  and  $|V_f|$ .

In the following we divide the problem of maximizing Equation 1 into two parts:

1. Determining a set of active lexicon entries.
2. Choosing the translation probabilities for the given set of active lexicon entries.

The second task can be achieved by running the EM algorithm on the restricted translation model. We deal with the first task in the following section.

#### 4 Monolingual Context Similarity

As described in Section 3 we need some mechanism to iteratively choose an active set of translation candidates. Based on the assumption that some of the active candidates and their respective probabilities are already correct, we induce new active candidates. In the context of information retrieval, Salton et al. (1975) introduce a document space where each

document identified by one or more index terms is represented by a high dimensional vector of term weights. Given two vectors  $v_1$  and  $v_2$  of two documents it is then possible to calculate a similarity coefficient between those given documents (which is usually denoted as  $s(v_1, v_2)$ ). Similar to this we represent source and target words in a high dimensional vector space of target word weights which we call *context vectors* and use a similarity coefficient to find possible translation pairs. We first initialize these context vectors using the following procedure:

1. Using only the monolingual data for the target language, prepare the context vectors  $v_{e_i}$  with entries  $v_{e_i, e_j}$ :
  - (a) Initialize all  $v_{e_i, e_j} = 0$
  - (b) For each target sentence  $E$ :
    - For each word  $e_i$  in  $E$ :
    - For each word  $e_j \neq e_i$  in  $E$ :
$$v_{e_i, e_j} = v_{e_i, e_j} + 1.$$
  - (c) Normalize each vector  $v_{e_i}$  such that  $\sum_{e_j} (v_{e_i, e_j})^2 \stackrel{!}{=} 1$  holds.

Using the notation  $\underline{e}_i = (e_j : v_{e_i, e_j}, \dots)$  these vectors might for example look like

$$\underline{work} = (early : 0.2, late : 0.1, \dots)$$

$$\underline{time} = (early : 0.2, late : 0.2, \dots).$$

2. Prepare context vectors  $v_{f_i, e_j}$  for the source language using only the monolingual data for the source language and the translation model's current parameter estimate  $\theta$ :
  - (a) Initialize all  $v_{f_i, e_j} = 0$
  - (b) Let  $\tilde{E}_\theta(F)$  denote the most probable translation of the foreign sentence  $F$  obtained by using the current estimate  $\theta$ .
  - (c) For each source sentence  $F$ :
    - For each word  $f_i$  in  $F$ :
    - For each word  $e_j \neq E_\theta(f_i)$ <sup>1</sup> in  $E_\theta(F)$ :
$$v_{f_i, e_j} = v_{f_i, e_j} + 1$$
  - (d) Normalize each vector  $v_{f_i}$  such that  $\sum_{e_j} (v_{f_i, e_j})^2 \stackrel{!}{=} 1$  holds.

<sup>1</sup>denoting that  $e_j$  is not the translation of  $f_i$  in  $E_\theta(F)$

Adapting the notation described above, these vectors might for example look like

$$\begin{aligned} \underline{Arbeit} &= (\text{early} : 0.25, \text{late} : 0.05, \dots) \\ \underline{Zeit} &= (\text{early} : 0.15, \text{late} : 0.25, \dots) \end{aligned}$$

Once we have set up the context vectors  $v_e$  and  $v_f$ , we can retrieve translation candidates for some source word  $f$  by finding those words  $e'$  that maximize the similarity coefficient  $s(v_{e'}, v_f)$ , as well as candidates for a given target word  $e$  by finding those words  $f'$  that maximize  $s(v_e, v_{f'})$ . In our implementation we use the Euclidean distance

$$d(v_e, v_f) = \|v_e - v_f\|_2. \quad (2)$$

as *distance* measure.<sup>2</sup> The normalization of context vectors described above is motivated by the fact that the context vectors should be invariant with respect to the absolute number of occurrences of words.<sup>3</sup>

Instead of just finding the best candidates for a given word, we are interested in an assignment that involves all source and target words, minimizing the sum of distances between the assigned words. In case of a one-to-one mapping the problem of assigning translation candidates such that the sum of distances is minimal can be solved optimally in polynomial time using the *hungarian algorithm* (Kuhn, 1955). In our case we are dealing with a many-to-many assignment that needs to satisfy the *maximum number of candidates* constraints. For this, we solve the problem in a greedy fashion by simply choosing the best pairs  $(e, f)$  first. As soon as a target word  $e$  or source word  $f$  has reached the limit of maximum candidates, we skip all further candidates for that word  $e$  (or  $f$  respectively). This step involves calculating and sorting all  $|V_e| \cdot |V_f|$  distances which can be done in time  $\mathcal{O}(V^2 \cdot \log(V))$ , with  $V = \max(|V_e|, |V_f|)$ . A simplified example of this procedure is depicted in Figure 1. The example already shows that the assignment obtained by this algorithm is in general not optimal.

<sup>2</sup>We then obtain pairs  $(e, f)$  that *minimize*  $d$ .

<sup>3</sup>This gives the same similarity ordering as using unnormalized vectors with the *cosine similarity measure*  $\frac{v_e \cdot v_f}{\|v_e\|_2 \cdot \|v_f\|_2}$  which can be interpreted as measuring the cosine of the angle between the vectors, see (Manning et al., 2008). Still it is noteworthy that this procedure is not equivalent to the *tf-IDF* context vectors described in (Salton et al., 1975).

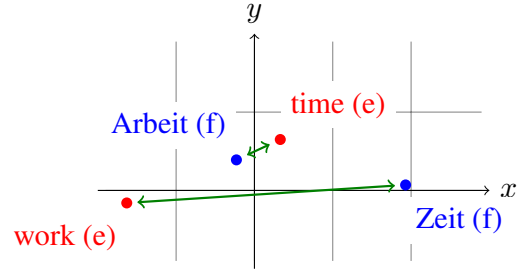


Figure 1: Hypothetical example for a greedy one-to-one assignment of translation candidates. The optimal assignment would contain (time,Zeit) and (work,Arbeit).

## 5 Training Algorithm and Implementation

Given the model presented in Section 3 and the methods illustrated in Section 4, we now describe how to train this model.

As described in Section 4, the overall procedure is divided into two alternating steps: After initialization we first perform EM training of the translation model for 20-30 iterations using a 2-gram or 3-gram language model in the target language. With the obtained best translations we induce new translation candidates using context similarity. This procedure is depicted in Figure 2.

### 5.1 Initialization

Let  $N_C$  be the maximum number of candidates per source word we allow,  $V_e$  and  $V_f$  be the target/source vocabulary and  $r(e)$  and  $r(f)$  the frequency rank of a source/target word. Each word  $f \in V_f$  with frequency rank  $r(f)$  is assigned to all words  $e \in V_e$  with frequency rank

$$r(e) \in [\text{start}(f), \text{end}(f)] \quad (3)$$

where

$$\text{start}(f) = \max(0, \min(|V_e| - N_c, \left\lfloor \frac{|V_e|}{|V_f|} \cdot r(f) - \frac{N_c}{2} \right\rfloor)) \quad (4)$$

$$\text{end}(f) = \min(\text{start}(f) + N_c, |V_e|). \quad (5)$$

This defines a diagonal beam<sup>4</sup> when visualizing the lexicon entries in a matrix where both source and target words are sorted by their frequency rank. However, note that the result of sorting by frequency

<sup>4</sup>The diagonal has some artifacts for the highest and lowest frequency ranks. See, for example, left side of Figure 2.

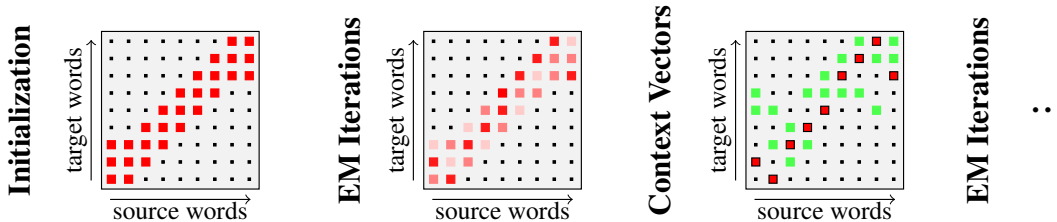


Figure 2: Visualization of the training procedure. The big rectangles represent word lexica in different stages of the training procedure. The small rectangles represent word pairs  $(e, f)$  for which  $e$  is a translation candidate of  $f$ , while dots represent word pairs  $(e, f)$  for which this is not the case. Source and target words are sorted by frequency so that the most frequent source words appear on the very left, and the most frequent target words appear at the very bottom.

and thus the frequency ranks are not unique when there are words with the same frequency. In this case, we initially obtain some not further specified frequency ordering, which is then kept throughout the procedure.

This initialization proves useful as we show by taking an IBM1 lexicon  $P(f|e)$  extracted on the parallel VERBMOBIL corpus (Wahlster, 2000): For each word  $e$  we calculate the weighted rank difference

$$\Delta r_{\text{avg}}(e) = \sum_f P(f|e) \cdot |(r(e) - r(f))| \quad (6)$$

and count how many of those weighted rank differences are smaller than a given value  $\frac{N_C}{2}$ . Here we see that for about 1% of the words the weighted rank difference lies within  $N_C = 50$ , and even about 3% for  $N_C = 150$  respectively. This shows that the initialization provides a first solid guess of possible translations.

## 5.2 EM Algorithm

The generative story described in Section 3 is implemented as a cascade of a permutation, insertion, lexicon, deletion and language model finite state transducers using OpenFST (Allauzen et al., 2007). Our FST representation of the LM makes use of failure transitions as described in (Allauzen et al., 2003). We use the forward-backward algorithm on the composed transducers to efficiently train the lexicon model using the EM algorithm.

## 5.3 Context Vector Step

Given the trained parameters  $\theta$  from the previous run of the EM algorithm we set the context vectors  $v_e$

and  $v_f$  up as described in Section 4. We then calculate and sort all  $|V_e| \cdot |V_f|$  distances which proves feasible in a few CPU hours even for vocabulary sizes of more than 50,000 words. This is achieved with the GNU SORT tool, which uses external sorting for sorting large amounts of data.

To set up the new lexicon we keep the  $\lfloor \frac{N_C}{2} \rfloor$  best translations for each source word with respect to  $P(e|f)$ , which we obtained in the previous EM run. Experiments showed that it is helpful to also limit the number of candidates per target words. We therefore prune the resulting lexicon using  $P(f|e)$  to a maximum of  $\lfloor \frac{N'_C}{2} \rfloor$  candidates per target word afterwards. Then we fill the lexicon with new candidates using the previously sorted list of candidate pairs such that the final lexicon has at most  $N_C$  candidates per source word and at most  $N'_C$  candidates per target word. We set  $N'_C$  to some value  $N'_C > N_C$ . All experiments in this work were run with  $N'_C = 300$ . Values of  $N'_C \approx N_C$  seem to produce poorer results. Not limiting the number of candidates per target word at all also typically results in weaker performance. After the lexicon is filled with candidates, we initialize the probabilities to be uniform. With this new lexicon the process is iterated starting with the EM training.

## 6 Experimental Evaluation

We evaluate our method on three different corpora.

At first we apply our method to non-parallel Spanish/English data that is based on the OPUS corpus (Tiedemann, 2009) and that was also used in (Ravi and Knight, 2011). We show that our method performs better by 1.6 BLEU than the best performing method described in (Ravi and Knight, 2011) while

Name	Lang.	Sent.	Words	Voc.
OPUS	Spanish	13,181	39,185	562
	English	19,770	61,835	411
VERBMOBIL	German	27,861	282,831	5,964
	English	27,862	294,902	3,723
GIGAWORD	French	100,000	1,725,993	68,259
	English	100,000	1,788,025	64,621

Table 1: Statistics of the corpora used in this paper.

being approximately 15 to 20 times faster than their  $n$ -gram based approach.

After that we apply our method to a non-parallel version of the German/English VERBMOBIL corpus, which has a vocabulary size of 6,000 words on the German side, and 3,500 words on the target side and which thereby is approximately one order of magnitude larger than the previous OPUS experiment.

We finally run our system on a subset of the non-parallel French/English GIGAWORD corpus, which has a vocabulary size of 60,000 words for both French and English. We show first interesting results on such a big task.

In case of the OPUS and VERBMOBIL corpus, we evaluate the results using BLEU (Papineni et al., 2002) and TER (Snover et al., 2006) to reference translations. We report all scores in percent. For BLEU higher values are better, for TER lower values are better. We also compare the results on these corpora to a system trained on parallel data.

In case of the GIGAWORD corpus we show lexicon entries obtained during training.

## 6.1 OPUS Subtitle Corpus

### 6.1.1 Experimental Setup

We apply our method to the corpus described in Table 6. This exact corpus was also used in (Ravi and Knight, 2011). The best performing methods in (Ravi and Knight, 2011) use the full  $411 \times 579$  lexicon model and apply standard EM training. Using a 2-gram LM they obtain 15.3 BLEU and with a whole segment LM, they achieve 19.3 BLEU. In comparison to this baseline we run our algorithm with  $N_C = 50$  candidates per source word for both, a 2-gram and a 3-gram LM. We use 30 EM iterations

between each context vector step. For both cases we run 7 EM+Context cycles.

### 6.1.2 Results

Figure 3 and Figure 4 show the evolution of BLEU and TER scores for applying our method using a 2-gram and a 3-gram LM.

In case of the 2-gram LM (Figure 3) the translation quality increases until it reaches a plateau after 5 EM+Context cycles. In case of the 3-gram LM (Figure 4) the statement only holds with respect to TER. It is notable that during the first iterations TER only improves very little until a large chunk of the language unravels after the third iteration. This behavior may be caused by the fact that the corpus only provides a relatively small amount of context information for each word, since sentence lengths are 3-4 words on average.

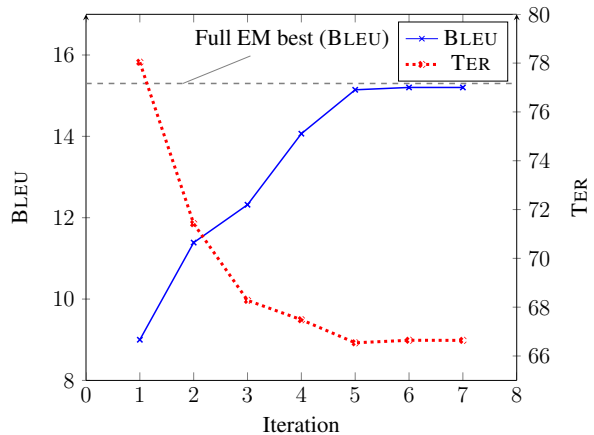


Figure 3: Results on the OPUS corpus with a 2-gram LM,  $N_C = 50$ , and 30 EM iterations between each context vector step. The dashed line shows the best result using a 2-gram LM in (Ravi and Knight, 2011).

Table 2 summarizes these results and compares them with (Ravi and Knight, 2011). Our 3-gram based method performs by 1.6 BLEU better than their best system which is a statistically significant improvement at 95% confidence level. Furthermore, Table 2 compares the CPU time needed for training. Our 3-gram based method is 15-20 times faster than running the EM based training procedure presented in (Ravi and Knight, 2011) with a 3-gram LM<sup>5</sup>.

<sup>5</sup>(Ravi and Knight, 2011) only report results using a 2-gram LM and a whole-segment LM.

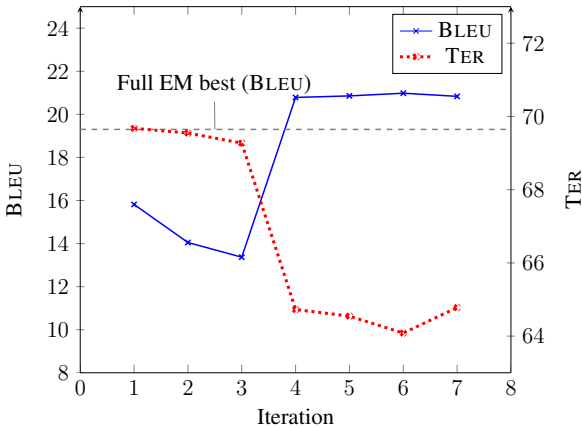


Figure 4: Results on the OPUS corpus with a 3-gram LM,  $N_C = 50$ , and 30 EM iterations between each context vector step. The dashed line shows the best result using a whole-segment LM in (Ravi and Knight, 2011)

Method	CPU	BLEU	TER
EM, 2-gram LM 411 cand. p. source word (Ravi and Knight, 2011)	$\approx 850h^6$	15.3	—
EM, Whole-segment LM 411 cand. p. source word (Ravi and Knight, 2011)	$-^7$	19.3	—
EM+Context, 2-gram LM 50 cand. p. source word (this work)	<b>50h<sup>8</sup></b>	<b>15.2</b>	<b>66.6</b>
EM+Context, 3-gram LM 50 cand. p. source word (this work)	<b>200h<sup>8</sup></b>	<b>20.9</b>	<b>64.5</b>

Table 2: Results obtained on the OPUS corpus.

To summarize: Our method is significantly faster than  $n$ -gram LM based approaches and obtains better results than any previously published method.

<sup>6</sup>Estimated by running full EM using the 2-gram LM using our implementation for 90 Iterations yielding 15.2 BLEU.

<sup>7</sup> $\approx 4,000h$  when running full EM using a 3-gram LM, using our implementation. Estimated by running only the first iteration and by assuming that the final result will be obtained after 90 iterations. However, (Ravi and Knight, 2011) report results using a whole segment LM, assigning  $P(e) > 0$  only to sequences seen in training. This seems to work for the given task but we believe that it can not be a general replacement for higher order  $n$ -gram LMs.

<sup>8</sup>Estimated by running our method for  $5 \times 30$  iterations.

## 6.2 VERBMOBIL Corpus

### 6.2.1 Experimental Setup

The VERBMOBIL corpus is a German/English corpus dealing with short sentences for making appointments. We prepared a non-parallel subset of the original VERBMOBIL (Wahlster, 2000) by splitting the corpus into two parts and then selecting only the German side from the first half, and the English side from the second half such that the target side is not the translation of the source side. The source and target vocabularies of the resulting non-parallel corpus are both more than 9 times bigger compared to the OPUS vocabularies. Also the total amount of word tokens is more than 5 times larger compared to the OPUS corpus. Table 6 shows the statistics of this corpus. We run our method for 5 EM+Context cycles (30 EM iterations each) using a 2-gram LM. After that we run another five EM+Context cycles using a 3-gram LM.

### 6.2.2 Results

Our results on the VERBMOBIL corpus are summarized in Table 3. Even on this more complex task our method achieves encouraging results: The

Method	BLEU	TER
$5 \times 30$ Iterations EM+Context 50 cand. p. source word, 2-gram LM	<b>11.7</b>	<b>67.4</b>
$+ 5 \times 30$ Iterations EM+Context 50 cand. p. source word, 3-gram LM	<b>15.5</b>	<b>63.2</b>

Table 3: Results obtained on the VERBMOBIL corpus.

translation quality increases from iteration to iteration until the algorithm finally reaches 11.7 BLEU using only the 2-gram LM. Running further five cycles using a 3-gram LM achieves a final performance of 15.5 BLEU. Och (2002) reports results of 48.2 BLEU for a single-word based translation system and 56.1 BLEU using the alignment template approach, both trained on parallel data. However, it should be noted that our experiment only uses 50% of the original VERBMOBIL training data to simulate a truly non-parallel setup.

Iter.	$e$	$p(f_1 e)$	$f_1$	$p(f_2 e)$	$f_2$	$p(f_3 e)$	$f_3$	$p(f_4 e)$	$f_4$	$p(f_5 e)$	$f_5$
1.	<b>the</b>	0.43	<i>la</i>	0.31	<i>l'</i>	0.11	<i>une</i>	0.04	<i>le</i>	0.04	<i>les</i>
2.	<b>several</b>	0.57	<i>plusieurs</i>	0.21	<i>les</i>	0.09	<i>des</i>	0.03	<i>nombreuses</i>	0.02	<i>deux</i>
3.	<b>where</b>	0.63	<i>où</i>	0.17	<i>mais</i>	0.06	<i>indique</i>	0.04	<i>précise</i>	0.02	<i>appelle</i>
4.	<b>see</b>	0.49	<i>éviter</i>	0.09	<i>effet</i>	0.09	<i>voir</i>	0.05	<i>envisager</i>	0.04	<i>dire</i>
5.	<b>January</b>	0.25	<i>octobre</i>	0.22	<i>mars</i>	0.09	<i>juillet</i>	0.07	<i>août</i>	0.07	<i>janvier</i>
–	<b>Germany</b>	0.24	<i>Italie</i>	0.12	<i>Espagne</i>	0.06	<i>Japon</i>	0.05	<i>retour</i>	0.05	<i>Suisse</i>

Table 4: Lexicon entries obtained by running our method on the non-parallel GIGAWORD corpus. The first column shows in which iteration the algorithm found the first correct translations  $f$  (compared to a parallelly trained lexicon) among the top 5 candidates

## 6.3 GIGAWORD

### 6.3.1 Experimental Setup

This setup is based on a subset of the monolingual GIGAWORD corpus. We selected 100,000 French sentences from the news agency *AFP* and 100,000 sentences from the news agency *Xinhua*. To have a more reliable set of training instances, we selected only sentences with more than 7 tokens. Note that these corpora form true non-parallel data which, besides the length filtering, were not specifically pre-selected or pre-processed. More details on these non-parallel corpora are summarized in Table 6. The vocabularies have a size of approximately 60,000 words which is more than 100 times larger than the vocabularies of the OPUS corpus. Also it incorporates more than 25 times as many tokens as the OPUS corpus.

After initialization, we run our method with  $N_C = 150$  candidates per source word for 20 EM iterations using a 2-gram LM. After the first context vector step with  $N_C = 50$  we run another  $4 \times 20$  iterations with  $N_C = 50$  with a 2-gram LM.

### 6.3.2 Results

Table 4 shows example lexicon entries we obtained. Note that we obtained these results by using purely non-parallel data, and that we neither used a seed lexicon, nor orthographic features to assign e.g. numbers or proper names: All results are obtained using 2-gram statistics and the context of words only. We find the results encouraging and think that they show the potential of large-scale unsupervised techniques for MT in the future.

## 7 Conclusion

We presented a method for learning statistical machine translation models from non-parallel data. The key to our method lies in limiting the translation model to a limited set of translation candidates and then using the EM algorithm to learn the probabilities. Based on the translations obtained with this model we obtain new translation candidates using a context vector approach. This method increased the training speed by a factor of 10-20 compared to methods known in literature and also resulted in a 1.6 BLEU point increase compared to previous approaches. Due to this efficiency improvement we were able to tackle larger tasks, such as a non-parallel version of the VERBMOBIL corpus having a nearly 10 times larger vocabulary. We also had a look at first results of our method on an even larger Task, incorporating a vocabulary of 60,000 words. We have shown that, using a limited set of translation candidates, we can significantly reduce the computational complexity of the learning task. This work serves as a big step towards large-scale unsupervised training for statistical machine translation systems.

## Acknowledgements

This work was realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation. The authors would like to thank Sujith Ravi and Kevin Knight for providing us with the OPUS subtitle corpus and David Rybach for kindly sharing his knowledge about the OpenFST library.



## References

- Cyril Allauzen, Mehryar Mohri, and Brian Roark. 2003. Generalized algorithms for constructing statistical language models. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 40–47. Association for Computational Linguistics.
- Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. Openfst: A general and efficient weighted finite-state transducer library. In Jan Holub and Jan Zdárek, editors, *CIAA*, volume 4783 of *Lecture Notes in Computer Science*, pages 11–23. Springer.
- Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, B*, 39.
- Aria Haghighi, Percy Liang, T Berg-Kirkpatrick, and Dan Klein. 2008. Learning Bilingual Lexicons from Monolingual Corpora. In *Proceedings of ACL08 HLT*, pages 771–779. Association for Computational Linguistics.
- Kevin Knight and Kenji Yamada. 1999. A computational approach to deciphering unknown scripts. In *ACL Workshop on Unsupervised Learning in Natural Language Processing*, number 1, pages 37–44. Cite-seer.
- Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL02 workshop on Unsupervised lexical acquisition*, number July, pages 9–16. Association for Computational Linguistics.
- Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83–97.
- Shou-de Lin and Kevin Knight. 2006. Discovering the linear writing order of a two-dimensional ancient hieroglyphic script. *Artificial Intelligence*, 170:409–421, April.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schuetze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, 1 edition, July.
- Franz J. Och. 2002. *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, RWTH Aachen University, Aachen, Germany, October.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL '02*, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 812–819, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Sujith Ravi and Kevin Knight. 2011. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Gerard M. Salton, Andrew K. C. Wong, and Chang S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, November.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA, August.
- Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *48th Annual Meeting of the Association for Computational Linguistics*, number July, pages 1048–1057.
- Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.
- Wolfgang Wahlster, editor. 2000. *Verbmobil: Foundations of speech-to-speech translations*. Springer-Verlag, Berlin.



# Machine Translation without Words through Substring Alignment

Graham Neubig<sup>1,2</sup>, Taro Watanabe<sup>2</sup>, Shinsuke Mori<sup>1</sup>, Tatsuya Kawahara<sup>1</sup>

<sup>1</sup>Graduate School of Informatics, Kyoto University  
Yoshida Honmachi, Sakyo-ku, Kyoto, Japan

<sup>2</sup>National Institute of Information and Communication Technology  
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan

## Abstract

In this paper, we demonstrate that accurate machine translation is possible without the concept of “words,” treating MT as a problem of transformation between character strings. We achieve this result by applying phrasal inversion transduction grammar alignment techniques to character strings to train a character-based translation model, and using this in the phrase-based MT framework. We also propose a look-ahead parsing algorithm and substring-informed prior probabilities to achieve more effective and efficient alignment. In an evaluation, we demonstrate that character-based translation can achieve results that compare to word-based systems while effectively translating unknown and uncommon words over several language pairs.

## 1 Introduction

Traditionally, the task of statistical machine translation (SMT) is defined as translating a source sentence  $f_1^J = \{f_1, \dots, f_J\}$  to a target sentence  $e_1^I = \{e_1, \dots, e_I\}$ , where each element of  $f_1^J$  and  $e_1^I$  is assumed to be a word in the source and target languages. However, the definition of a “word” is often problematic. The most obvious example of this lies in languages that do not separate words with white space such as Chinese, Japanese, or Thai, in which the choice of a segmentation standard has a large effect on translation accuracy (Chang et al., 2008). Even for languages with explicit word

The first author is now affiliated with the Nara Institute of Science and Technology.

boundaries, all machine translation systems perform at least some precursory form of tokenization, splitting punctuation and words to prevent the sparsity that would occur if punctuated and non-punctuated words were treated as different entities. Sparsity also manifests itself in other forms, including the large vocabularies produced by morphological productivity, word compounding, numbers, and proper names. A myriad of methods have been proposed to handle each of these phenomena individually, including morphological analysis, stemming, compound breaking, number regularization, optimizing word segmentation, and transliteration, which we outline in more detail in Section 2.

These difficulties occur because we are translating sequences of *words* as our basic unit. On the other hand, Vilar et al. (2007) examine the possibility of instead treating each sentence as sequences of *characters* to be translated. This method is attractive, as it is theoretically able to handle all sparsity phenomena in a single unified framework, but has only been shown feasible between similar language pairs such as Spanish-Catalan (Vilar et al., 2007), Swedish-Norwegian (Tiedemann, 2009), and Thai-Lao (Sornlertlamvanich et al., 2008), which have a strong co-occurrence between single characters. As Vilar et al. (2007) state and we confirm, accurate translations cannot be achieved when applying traditional translation techniques to character-based translation for less similar language pairs.

In this paper, we propose improvements to the alignment process tailored to character-based machine translation, and demonstrate that it is, in fact, possible to achieve translation accuracies that ap-

proach those of traditional word-based systems using only character strings. We draw upon recent advances in many-to-many alignment, which allows for the automatic choice of the length of units to be aligned. As these units may be at the character, subword, word, or multi-word phrase level, we conjecture that this will allow for better character alignments than one-to-many alignment techniques, and will allow for better translation of uncommon words than traditional word-based models by breaking down words into their component parts.

We also propose two improvements to the many-to-many alignment method of Neubig et al. (2011). One barrier to applying many-to-many alignment models to character strings is training cost. In the inversion transduction grammar (ITG) framework (Wu, 1997), which is widely used in many-to-many alignment, search is cumbersome for longer sentences, a problem that is further exacerbated when using characters instead of words as the basic unit. As a step towards overcoming this difficulty, we increase the efficiency of the beam-search technique of Saers et al. (2009) by augmenting it with look-ahead probabilities in the spirit of A\* search. Secondly, we describe a method to seed the search process using counts of all substring pairs in the corpus to bias the phrase alignment model. We do this by defining prior probabilities based on these substring counts within the Bayesian phrasal ITG framework.

An evaluation on four language pairs with differing morphological properties shows that for distant language pairs, character-based SMT can achieve translation accuracy comparable to word-based systems. In addition, we perform ablation studies, showing that these results were not possible without the proposed enhancements to the model. Finally, we perform a qualitative analysis, which finds that character-based translation can handle unsegmented text, conjugation, and proper names in a unified framework with no additional processing.

## 2 Related Work on Data Sparsity in SMT

As traditional SMT systems treat all words as single tokens without considering their internal structure, major problems of data sparsity occur for less frequent tokens. In fact, it has been shown that there is a direct negative correlation between vocabulary

size (and thus sparsity) of a language and translation accuracy (Koehn, 2005). Sparsity causes trouble for alignment models, both in the form of incorrectly aligned uncommon words, and in the form of garbage collection, where uncommon words in one language are incorrectly aligned to large segments of the sentence in the other language (Och and Ney, 2003). Unknown words are also a problem during the translation process, and the default approach is to map them as-is into the target sentence.

This is a major problem in agglutinative languages such as Finnish or compounding languages such as German. Previous works have attempted to handle morphology, decompounding and regularization through lemmatization, morphological analysis, or unsupervised techniques (Nießen and Ney, 2000; Brown, 2002; Lee, 2004; Goldwater and McClosky, 2005; Talbot and Osborne, 2006; Mermer and Akin, 2010; Macherey et al., 2011). It has also been noted that it is more difficult to translate into morphologically rich languages, and methods for modeling target-side morphology have attracted interest in recent years (Bojar, 2007; Subotin, 2011).

Another source of data sparsity that occurs in all languages is proper names, which have been handled by using cognates or transliteration to improve translation (Knight and Graehl, 1998; Kondrak et al., 2003; Finch and Sumita, 2007), and more sophisticated methods for named entity translation that combine translation and transliteration have also been proposed (Al-Onaizan and Knight, 2002).

Choosing word units is also essential for creating good translation results for languages that do not explicitly mark word boundaries, such as Chinese, Japanese, and Thai. A number of works have dealt with this word segmentation problem in translation, mainly focusing on Chinese-to-English translation (Bai et al., 2008; Chang et al., 2008; Zhang et al., 2008b; Chung and Gildea, 2009; Nguyen et al., 2010), although these works generally assume that a word segmentation exists in one language (English) and attempt to optimize the word segmentation in the other language (Chinese).

We have enumerated these related works to demonstrate the myriad of data sparsity problems and proposed solutions. Character-based translation has the potential to handle all of the phenomena in the previously mentioned research in a single

unified framework, requiring no language specific tools such as morphological analyzers or word segmenters. However, while the approach is attractive conceptually, previous research has only been shown effective for closely related language pairs (Vilar et al., 2007; Tiedemann, 2009; Sornlertlamvanich et al., 2008). In this work, we propose effective alignment techniques that allow character-based translation to achieve accurate translation results for both close and distant language pairs.

### 3 Alignment Methods

SMT systems are generally constructed from a parallel corpus consisting of target language sentences  $\mathcal{E}$  and source language sentences  $\mathcal{F}$ . The first step of training is to find alignments  $\mathcal{A}$  for the words in each sentence pair.

We represent our target and source sentences as  $e_1^I$  and  $f_1^J$ .  $e_i$  and  $f_j$  represent single elements of the target and source sentences respectively. These may be words in word-based alignment models or single characters in character-based alignment models.<sup>1</sup> We define our alignment as  $\mathbf{a}_1^K$ , where each element is a span  $a_k = \langle s, t, u, v \rangle$  indicating that the target string  $e_s, \dots, e_t$  and source string  $f_u, \dots, f_v$  are aligned to each-other.

#### 3.1 One-to-Many Alignment

The most well-known and widely-used models for bitext alignment are for one-to-many alignment, including the IBM models (Brown et al., 1993) and HMM alignment model (Vogel et al., 1996). These models are by nature directional, attempting to find the alignments that maximize the conditional probability of the target sentence  $P(e_1^I | f_1^J, \mathbf{a}_1^K)$ . For computational reasons, the IBM models are restricted to aligning each word on the target side to a single word on the source side. In the formalism presented above, this means that each  $e_i$  must be included in at most one span, and for each span  $u = v$ . Traditionally, these models are run in both directions and combined using heuristics to create many-to-many alignments (Koehn et al., 2003).

However, in order for one-to-many alignment methods to be effective, each  $f_j$  must contain

<sup>1</sup>Some previous work has also performed alignment using morphological analyzers to normalize or split the sentence into morpheme streams (Corston-Oliver and Gamon, 2004).

enough information to allow for effective alignment with its corresponding elements in  $e_1^I$ . While this is often the case in word-based models, for character-based models this assumption breaks down, as there is often no clear correspondence between characters.

#### 3.2 Many-to-Many Alignment

On the other hand, in recent years, there have been advances in many-to-many alignment techniques that are able to align multi-element chunks on both sides of the translation (Marcu and Wong, 2002; DeNero et al., 2008; Blunsom et al., 2009; Neubig et al., 2011). Many-to-many methods can be expected to achieve superior results on character-based alignment, as the aligner can use information about substrings, which may correspond to letters, morphemes, words, or short phrases.

Here, we focus on the model presented by Neubig et al. (2011), which uses Bayesian inference in the phrasal inversion transduction grammar (ITG, Wu (1997)) framework. ITGs are a variety of synchronous context free grammar (SCFG) that allows for many-to-many alignment to be achieved in polynomial time through the process of biparsing, which we explain more in the following section. Phrasal ITGs are ITGs that allow for non-terminals that can emit phrase pairs with multiple elements on both the source and target sides. It should be noted that there are other many-to-many alignment methods that have been used for simultaneously discovering morphological boundaries over multiple languages (Snyder and Barzilay, 2008; Naradowsky and Toutanova, 2011), but these have generally been applied to single words or short phrases, and it is not immediately clear that they will scale to aligning full sentences.

### 4 Look-Ahead Biparsing

In this work, we experiment with the alignment method of Neubig et al. (2011), which can achieve competitive accuracy with a much smaller phrase table than traditional methods. This is important in the character-based translation context, as we would like to use phrases that contain large numbers of characters without creating a phrase table so large that it cannot be used in actual decoding. In this framework, training is performed using sentence-

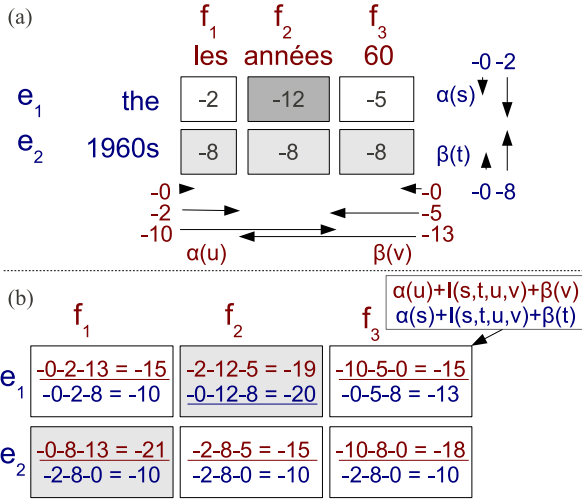


Figure 1: (a) A chart with inside probabilities in boxes and forward/backward probabilities marking the surrounding arrows. (b) Spans with corresponding look-aheads added, and the minimum probability underlined. Lightly and darkly shaded spans will be trimmed when the beam is  $\log(P) \geq -3$  and  $\log(P) \geq -6$  respectively.

wise block sampling, acquiring a sample for each sentence by first performing bottom-up biparsing to create a chart of probabilities, then performing top-down sampling of a new tree based on the probabilities in this chart.

An example of a chart used in this parsing can be found in Figure 1 (a). Within each cell of the chart spanning  $e_s^t$  and  $f_u^v$  is an “inside” probability  $I(a_{s,t,u,v})$ . This probability is the combination of the generative probability of each phrase pair  $P_t(e_s^t, f_u^v)$  as well as the sum the probabilities over all shorter spans in straight and inverted order<sup>2</sup>

$$\begin{aligned}
 I(a_{s,t,u,v}) &= P_t(e_s^t, f_u^v) \\
 &+ \sum_{s \leq S \leq t} \sum_{u \leq U \leq v} P_x(\text{str}) I(a_{s,S,u,U}) I(a_{S,t,U,v}) \\
 &+ \sum_{s \leq S \leq t} \sum_{u \leq U \leq v} P_x(\text{inv}) I(a_{s,S,U,v}) I(a_{S,t,u,U})
 \end{aligned}$$

where  $P_x(\text{str})$  and  $P_x(\text{inv})$  are the probability of straight and inverted ITG productions.

While the exact calculation of these probabilities can be performed in  $O(n^6)$  time, where  $n$  is the

<sup>2</sup> $P_t$  can be specified according to Bayesian statistics as described by Neubig et al. (2011).

length of the sentence, this is impractical for all but the shortest sentences. Thus it is necessary to use methods to reduce the search space such as beam-search based chart parsing (Saers et al., 2009) or slice sampling (Blunsom and Cohn, 2010).<sup>3</sup>

In this section we propose the use of a look-ahead probability to increase the efficiency of this chart parsing. Taking the example of Saers et al. (2009), spans are pushed onto a different queue based on their size, and queues are processed in ascending order of size. Agendas can further be trimmed based on a histogram beam (Saers et al., 2009) or probability beam (Neubig et al., 2011) compared to the best hypothesis  $\hat{a}$ . In other words, we have a queue discipline based on the inside probability, and all spans  $a_k$  where  $I(a_k) < cI(\hat{a})$  are pruned.  $c$  is a constant describing the width of the beam, and a smaller constant probability will indicate a wider beam.

This method is insensitive to the existence of competing hypotheses when performing pruning. Figure 1 (a) provides an example of why it is unwise to ignore competing hypotheses during beam pruning. Particularly, the alignment “les/1960s” competes with the high-probability alignment “les/the,” so intuitively should be a good candidate for pruning. However its probability is only slightly higher than “années/1960s,” which has no competing hypotheses and thus should not be trimmed.

In order to take into account competing hypotheses, we can use for our queue discipline not only the inside probability  $I(a_k)$ , but also the outside probability  $O(a_k)$ , the probability of generating all spans other than  $a_k$ , as in A\* search for CFGs (Klein and Manning, 2003), and tic-tac-toe pruning for word-based ITGs (Zhang and Gildea, 2005). As the calculation of the actual outside probability  $O(a_k)$  is just as expensive as parsing itself, it is necessary to approximate this with heuristic function  $O^*$  that can be calculated efficiently.

Here we propose a heuristic function that is designed specifically for phrasal ITGs and is computable with worst-case complexity of  $n^2$ , compared with the  $n^3$  amortized time of the tic-tac-toe pruning

<sup>3</sup>Applying beam-search before sampling will sample from an improper distribution, although Metropolis-in-Gibbs sampling (Johnson et al., 2007) can be used to compensate. However, we found that this had no significant effect on results, so we omit the Metropolis-in-Gibbs step for experiments.

algorithm described by (Zhang et al., 2008a). During the calculation of the phrase generation probabilities  $P_t$ , we save the best inside probability  $I^*$  for each monolingual span.

$$I_e^*(s, t) = \max_{\{\tilde{u}=\langle\tilde{s},\tilde{t},\tilde{u},\tilde{v}\rangle;\tilde{s}=s,\tilde{t}=t\}} P_t(\tilde{a})$$

$$I_f^*(u, v) = \max_{\{\tilde{a}=\langle\tilde{s},\tilde{t},\tilde{u},\tilde{v}\rangle;\tilde{u}=u,\tilde{v}=v\}} P_t(\tilde{a})$$

For each language independently, we calculate forward probabilities  $\alpha$  and backward probabilities  $\beta$ . For example,  $\alpha_e(s)$  is the maximum probability of the span  $(0, s)$  of  $e$  that can be created by concatenating together consecutive values of  $I_e^*$ :

$$\alpha_e(s) = \max_{\{S_1, \dots, S_x\}} I_e^*(0, S_1) I_e^*(S_1, S_2) \dots I_e^*(S_x, s).$$

Backwards probabilities and probabilities over  $f$  can be defined similarly. These probabilities are calculated for  $e$  and  $f$  independently, and can be calculated in  $n^2$  time by processing each  $\alpha$  in ascending order, and each  $\beta$  in descending order in a fashion similar to that of the forward-backward algorithm. Finally, for any span, we define the outside heuristic as the minimum of the two independent look-ahead probabilities over each language

$$O^*(a_{s,t,u,v}) = \min(\alpha_e(s) * \beta_e(t), \alpha_f(u) * \beta_f(v)).$$

Looking again at Figure 1 (b), it can be seen that the relative probability difference between the highest probability span “les/the” and the spans “années/1960s” and “60/1960s” decreases, allowing for tighter beam pruning without losing these good hypotheses. In contrast, the relative probability of “les/1960s” remains low as it is in conflict with a high-probability alignment, allowing it to be discarded.

## 5 Substring Prior Probabilities

While the Bayesian phrasal ITG framework uses the previously mentioned phrase distribution  $P_t$  during search, it also allows for definition of a phrase pair prior probability  $P_{prior}(e_s^t, f_u^v)$ , which can efficiently seed the search process with a bias towards phrase pairs that satisfy certain properties. In this section, we overview an existing method used to calculate these prior probabilities, and also propose a new way to calculate priors based on substring co-occurrence statistics.

### 5.1 Word-based Priors

Previous research on many-to-many translation has used IBM model 1 probabilities to bias phrasal alignments so that phrases whose member words are good translations are also aligned. As a representative of this existing method, we adopt a base measure similar to that used by DeNero et al. (2008):

$$P_{m1}(e, f) = M_0(e, f) P_{pois}(|e|; \lambda) P_{pois}(|f|; \lambda)$$

$$M_0(e, f) = (P_{m1}(f|e) P_{uni}(e) P_{m1}(e|f) P_{uni}(f))^{\frac{1}{2}}.$$

$P_{pois}$  is the Poisson distribution with the average length parameter  $\lambda$ , which we set to 0.01.  $P_{m1}$  is the word-based (or character-based) Model 1 probability, which can be efficiently calculated using the dynamic programming algorithm described by Brown et al. (1993). However, for reasons previously stated in Section 3, these methods are less satisfactory when performing character-based alignment, as the amount of information contained in a character does not allow for proper alignment.

### 5.2 Substring Co-occurrence Priors

Instead, we propose a method for using raw substring co-occurrence statistics to bias alignments towards substrings that often co-occur in the entire training corpus. This is similar to the method of Cromieres (2006), but instead of using these co-occurrence statistics as a heuristic alignment criterion, we incorporate them as a prior probability in a statistical model that can take into account mutual exclusivity of overlapping substrings in a sentence.

We define this prior probability using three counts over substrings  $c(e)$ ,  $c(f)$ , and  $c(e, f)$ .  $c(e)$  and  $c(f)$  count the total number of sentences in which the substrings  $e$  and  $f$  occur respectively.  $c(e, f)$  is a count of the total number of sentences in which the substring  $e$  occurs on the target side, and  $f$  occurs on the source side. We perform the calculation of these statistics using enhanced suffix arrays, a data structure that can efficiently calculate all substrings in a corpus (Abouelhoda et al., 2004).<sup>4</sup>

While suffix arrays allow for efficient calculation of these statistics, storing all co-occurrence counts  $c(e, f)$  is an unrealistic memory burden for larger

<sup>4</sup>Using the open-source implementation esaxx <http://code.google.com/p/esaxx/>

corpora. In order to reduce the amount of memory used, we discount every count by a constant  $d$ , which we set to 5. This has a dual effect of reducing the amount of memory needed to hold co-occurrence counts by removing values for which  $c(e, f) < d$ , as well as preventing over-fitting of the training data. In addition, we heuristically prune values for which the conditional probabilities  $P(e|f)$  or  $P(f|e)$  are less than some fixed value, which we set to 0.1 for the reported experiments.

To determine how to combine  $c(e)$ ,  $c(f)$ , and  $c(e, f)$  into prior probabilities, we performed preliminary experiments testing methods proposed by previous research including plain co-occurrence counts, the Dice coefficient, and  $\chi$ -squared statistics (Cromieres, 2006), as well as a new method of defining substring pair probabilities to be proportional to bidirectional conditional probabilities

$$P_{cooc}(e, f) = P_{cooc}(e|f)P_{cooc}(f|e)/Z \\ = \left( \frac{c(e, f) - d}{c(f) - d} \right) \left( \frac{c(e, f) - d}{c(e) - d} \right) / Z$$

for all substring pairs where  $c(e, f) > d$  and where  $Z$  is a normalization term equal to

$$Z = \sum_{\{e, f; c(e, f) > d\}} P_{cooc}(e|f)P_{cooc}(f|e).$$

The experiments showed that the bidirectional conditional probability method gave significantly better results than all other methods, so we adopt this for the remainder of our experiments.

It should be noted that as we are using discounting, many substring pairs will be given zero probability according to  $P_{cooc}$ . As the prior is only supposed to bias the model towards good solutions and not explicitly rule out any possibilities, we linearly interpolate the co-occurrence probability with the one-to-many Model 1 probability, which will give at least some probability mass to all substring pairs

$$P_{prior}(e, f) = \lambda P_{cooc}(e, f) + (1 - \lambda)P_{m1}(e, f).$$

We put a Dirichlet prior ( $\alpha = 1$ ) on the interpolation coefficient  $\lambda$  and learn it during training.

## 6 Experiments

In order to test the effectiveness of character-based translation, we performed experiments over a variety of language pairs and experimental settings.

	de-en	fi-en	fr-en	ja-en
TM (en)	2.80M	3.10M	2.77M	2.13M
TM (other)	2.56M	2.23M	3.05M	2.34M
LM (en)	16.0M	15.5M	13.8M	11.5M
LM (other)	15.3M	11.3M	15.6M	11.9M
Tune (en)	58.7k	58.7k	58.7k	30.8k
Tune (other)	55.1k	42.0k	67.3k	34.4k
Test (en)	58.0k	58.0k	58.0k	26.6k
Test (other)	54.3k	41.4k	66.2k	28.5k

Table 1: The number of words in each corpus for TM and LM training, tuning, and testing.

### 6.1 Experimental Setup

We use a combination of four languages with English, using freely available data. We selected French-English, German-English, Finnish-English data from EuroParl (Koehn, 2005), with development and test sets designated for the 2005 ACL shared task on machine translation.<sup>5</sup> We also did experiments with Japanese-English Wikipedia articles from the Kyoto Free Translation Task (Neubig, 2011) using the designated training and tuning sets, and reporting results on the test set. These languages were chosen as they have a variety of interesting characteristics. French has some inflection, but among the test languages has the strongest one-to-one correspondence with English, and is generally considered easy to translate. German has many compound words, which must be broken apart to translate properly into English. Finnish is an agglutinative language with extremely rich morphology, resulting in long words and the largest vocabulary of the languages in EuroParl. Japanese does not have any clear word boundaries, and uses logographic characters, which contain more information than phonetic characters.

With regards to data preparation, the EuroParl data was pre-tokenized, so we simply used the tokenized data as-is for the training and evaluation of all models. For word-based translation in the Kyoto task, training was performed using the provided tokenization scripts. For character-based translation, no tokenization was performed, using the original text for both training and decoding. For both tasks, we selected as training data all sentences for which both

<sup>5</sup><http://statmt.org/wpt05/mt-shared-task>

	de-en	fi-en	fr-en	ja-en
GIZA-word	<b>24.58</b> / 64.28 / 30.43	20.41 / 60.01 / 27.89	<b>30.23</b> / <b>68.79</b> / 34.20	<b>17.95</b> / 56.47 / <b>24.70</b>
ITG-word	23.87 / <b>64.89</b> / <b>30.71</b>	<b>20.83</b> / 61.04 / 28.46	<b>29.92</b> / 68.64 / <b>34.29</b>	17.14 / 56.60 / <b>24.89</b>
GIZA-char	08.05 / 45.01 / 15.35	06.91 / 41.62 / 14.39	11.05 / 48.23 / 17.80	09.46 / 49.02 / 18.34
ITG-char	21.79 / 64.47 / 30.12	18.38 / <b>62.44</b> / <b>28.94</b>	26.70 / 66.76 / 32.47	15.84 / <b>58.41</b> / 24.58

	en-de	en-fi	en-fr	en-ja
GIZA-word	<b>17.94</b> / 62.71 / <b>37.88</b>	<b>13.22</b> / 58.50 / <b>27.03</b>	<b>32.19</b> / 69.20 / <b>52.39</b>	<b>20.79</b> / 27.01 / <b>38.41</b>
ITG-word	17.47 / <b>63.18</b> / <b>37.79</b>	<b>13.12</b> / <b>59.27</b> / <b>27.09</b>	31.66 / <b>69.61</b> / 51.98	<b>20.26</b> / <b>28.34</b> / <b>38.34</b>
GIZA-char	06.17 / 41.04 / 19.90	04.58 / 35.09 / 11.76	10.31 / 42.84 / 25.06	01.48 / 00.72 / 06.67
ITG-char	15.35 / 61.95 / 35.45	12.14 / <b>59.02</b> / 25.31	27.74 / 67.44 / 48.56	17.90 / <b>28.46</b> / 35.71

Table 2: Translation results in word-based BLEU, character-based BLEU, and METEOR for the GIZA++ and phrasal ITG models for word and character-based translation, with bold numbers indicating a statistically insignificant difference from the best system according to the bootstrap resampling method at  $p = 0.05$  (Koehn, 2004).

source and target were 100 characters or less,<sup>6</sup> the total size of which is shown in Table 1. In character-based translation, white spaces between words were treated as any other character and not given any special treatment. Evaluation was performed on tokenized and lower-cased data.

For alignment, we use the GIZA++ implementation of one-to-many alignment<sup>7</sup> and the pialign implementation of the phrasal ITG models<sup>8</sup> modified with the proposed improvements. For GIZA++, we used the default settings for word-based alignment, but used the HMM model for character-based alignment to allow for alignment of longer sentences. For pialign, default settings were used except for character-based ITG alignment, which used a probability beam of  $10^{-4}$  instead  $10^{-10}$ .<sup>9</sup> For decoding, we use the Moses decoder,<sup>10</sup> using the default settings except for the stack size, which we set to 1000 instead of 200. Minimum error rate training was performed to maximize word-based BLEU score for all systems.<sup>11</sup> For language models, word-based translation uses a word 5-gram model, and character-based translation uses a character 12-gram model, both smoothed using interpolated Kneser-Ney.

<sup>6</sup>100 characters is an average of 18.8 English words

<sup>7</sup><http://code.google.com/p/giza-pp/>

<sup>8</sup><http://phontron.com/pialign/>

<sup>9</sup>Improvement by using a beam larger than  $10^{-4}$  was marginal, especially with co-occurrence prior probabilities.

<sup>10</sup><http://statmt.org/moses/>

<sup>11</sup>We chose this set-up to minimize the effect of tuning criterion on our experiments, although it does indicate that we must have access to tokenized data for the development set.

## 6.2 Quantitative Evaluation

Table 2 presents a quantitative analysis of the translation results for each of the proposed methods. As previous research has shown that it is more difficult to translate into morphologically rich languages than into English (Koehn, 2005), we perform experiments translating in both directions for all language pairs. We evaluate translation quality using BLEU score (Papineni et al., 2002), both on the word and character level (with  $n = 4$ ), as well as METEOR (Denkowski and Lavie, 2011) on the word level.

It can be seen that character-based translation with all of the proposed alignment improvements greatly exceeds character-based translation using one-to-many alignment, confirming that substring-based information is necessary for accurate alignments. When compared with word-based translation, character-based translation achieves better, comparable, or inferior results on character-based BLEU, comparable or inferior results on METEOR, and inferior results on word-based BLEU. The differences between the evaluation metrics are due to the fact that character-based translation often gets words mostly correct other than one or two letters. These are given partial credit by character-based BLEU (and to a lesser extent METEOR), but marked entirely wrong by word-based BLEU.

Interestingly, for translation into English, character-based translation achieves higher accuracy compared to word-based translation on Japanese and Finnish input, followed by German,

	fi-en	ja-en
ITG-word	2.851	2.085
ITG-char	2.826	2.154

Table 3: Human evaluation scores (0-5 scale).

Source Unk. (13/26)	Ref: Word: Char:	directive on equality tasa-arvodirektiivi equality directive
Target Unk. (5/26)	Ref: Word: Char:	yoshiwara-juku station yoshiwara no eki yoshiwara-juku station
Uncommon (5/26)	Ref: Word: Char:	world health organisation world health world health organisation

Table 4: The major gains of character-based translation, unknown, hyphenated, and uncommon words.

and finally French. This confirms that character-based translation is performing well on languages that have long words or ambiguous boundaries, and less well on language pairs with relatively strong one-to-one correspondence between words.

### 6.3 Qualitative Evaluation

In addition, we performed a subjective evaluation of Japanese-English and Finnish-English translations. Two raters evaluated 100 sentences each, assigning a score of 0-5 based on how well the translation conveys the information contained in the reference. We focus on shorter sentences of 8-16 English words to ease rating and interpretation. Table 3 shows that the results are comparable, with no significant difference in average scores for either language pair.

Table 4 shows a breakdown of the sentences for which character-based translation received a score of at 2+ points more than word-based. It can be seen that character-based translation is properly handling sparsity phenomena. On the other hand, word-based translation was generally stronger with reordering and lexical choice of more common words.

### 6.4 Effect of Alignment Method

In this section, we compare the translation accuracies for character-based translation using the phrasal ITG model with and without the proposed improvements of substring co-occurrence priors and look-ahead parsing as described in Sections 4 and 5.2.

	fi-en	en-fi	ja-en	en-ja
ITG +cooc +look	<b>28.94</b>	<b>25.31</b>	<b>24.58</b>	<b>35.71</b>
ITG +cooc -look	28.51	24.24	<b>24.32</b>	<b>35.74</b>
ITG -cooc +look	<b>28.65</b>	24.49	<b>24.36</b>	35.05
ITG -cooc -look	27.45	23.30	23.57	34.50

Table 5: METEOR scores for alignment with and without look-ahead and co-occurrence priors.

Figure 5 shows METEOR scores<sup>12</sup> for experiments translating Japanese and Finnish. It can be seen that the co-occurrence prior gives gains in all cases, indicating that substring statistics are effectively seeding the ITG aligner. The introduced look-ahead probabilities improve accuracy significantly when substring co-occurrence counts are not used, and slightly when co-occurrence counts are used. More importantly, they allow for more aggressive beam pruning, increasing sampling speed from 1.3 sent/s to 2.5 sent/s for Finnish, and 6.8 sent/s to 11.6 sent/s for Japanese.

## 7 Conclusion and Future Directions

This paper demonstrated that character-based translation can act as a unified framework for handling difficult problems in translation: morphology, compound words, transliteration, and segmentation.

One future challenge includes scaling training up to longer sentences, which can likely be achieved through methods such as the heuristic span pruning of Haghighi et al. (2009) or sentence splitting of Vilar et al. (2007). Monolingual data could also be used to improve estimates of our substring-based prior. In addition, error analysis showed that word-based translation performed better than character-based translation on reordering and lexical choice, indicating that improved decoding (or pre-ordering) and language modeling tailored to character-based translation will likely greatly improve accuracy. Finally, we plan to explore the middle ground between word-based and character based translation, allowing for the flexibility of character-based translation, while using word boundary information to increase efficiency and accuracy.

<sup>12</sup>Similar results were found for character and word-based BLEU, but are omitted for lack of space.



## References

- Mohamed I. Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. 2004. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1).
- Yaser Al-Onaizan and Kevin Knight. 2002. Translating named entities using monolingual and bilingual resources. In *Proc. ACL*.
- Ming-Hong Bai, Keh-Jiann Chen, and Jason S. Chang. 2008. Improving word alignment by adjusting Chinese word segmentation. In *Proc. IJCNLP*.
- Phil Blunsom and Trevor Cohn. 2010. Inducing synchronous grammars with slice sampling. In *Proc. HLT-NAACL*, pages 238–241.
- Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A Gibbs sampler for phrasal synchronous grammar induction. In *Proc. ACL*.
- Ondřej Bojar. 2007. English-to-Czech factored machine translation. In *Proc. WMT*.
- Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19.
- Ralf D. Brown. 2002. Corpus-driven splitting of compound words. In *Proc. TMI*.
- Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proc. WMT*.
- Tagyoung Chung and Daniel Gildea. 2009. Unsupervised tokenization for machine translation. In *Proc. EMNLP*.
- Simon Corston-Oliver and Michael Gamon. 2004. Normalizing German and English inflectional morphology to improve statistical word alignment. *Machine Translation: From Real Users to Research*.
- Fabien Cromieres. 2006. Sub-sentential alignment using substring co-occurrence counts. In *Proc. COLING/ACL 2006 Student Research Workshop*.
- John DeNero, Alex Bouchard-Côté, and Dan Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *Proc. EMNLP*.
- Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proc. WMT*.
- Andrew Finch and Eiichiro Sumita. 2007. Phrase-based machine transliteration. In *Proc. TCAST*.
- Sharon Goldwater and David McClosky. 2005. Improving statistical MT through morphological analysis. In *Proc. EMNLP*.
- Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised ITG models. In *Proc. ACL*.
- Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Proc. NAACL*.
- Dan Klein and Christopher D. Manning. 2003. A\* parsing: fast exact Viterbi parse selection. In *Proc. HLT*.
- Kevin Knight and Jonathan Graehl. 1998. Machine transliteration. *Computational Linguistics*, 24(4).
- Phillip Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT*, pages 48–54.
- Phillip Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP*.
- Phillip Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.
- Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can improve statistical translation models. In *Proc. HLT*.
- Young-Suk Lee. 2004. Morphological analysis for statistical machine translation. In *Proc. HLT*.
- Klaus Macherey, Andrew Dai, David Talbot, Ashok Popat, and Franz Och. 2011. Language-independent compound splitting with morphological operations. In *Proc. ACL*.
- Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proc. EMNLP*.
- Coşkun Mermer and Ahmet Afşın Akın. 2010. Unsupervised search for the optimal segmentation for statistical machine translation. In *Proc. ACL Student Research Workshop*.
- Jason Naradowsky and Kristina Toutanova. 2011. Unsupervised bilingual morpheme segmentation and alignment with context-rich hidden semi-Markov models. In *Proc. ACL*.
- Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *Proc. ACL*, pages 632–641, Portland, USA, June.
- Graham Neubig. 2011. The Kyoto free translation task. <http://www.phontron.com/kftt>.
- ThuyLinh Nguyen, Stephan Vogel, and Noah A. Smith. 2010. Nonparametric word segmentation for machine translation. In *Proc. COLING*.
- Sonja Nießen and Hermann Ney. 2000. Improving SMT quality with morpho-syntactic analysis. In *Proc. COLING*.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. COLING*.

- Markus Saers, Joakim Nivre, and Dekai Wu. 2009. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *Proc. IWPT*, pages 29–32.
- Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. *Proc. ACL*.
- Virach Sornlertlamvanich, Chumpol Mokrat, and Hitoshi Isahara. 2008. Thai-lao machine translation based on phoneme transfer. In *Proc. 14th Annual Meeting of the Association for Natural Language Processing*.
- Michael Subotin. 2011. An exponential translation model for target language morphology. In *Proc. ACL*.
- David Talbot and Miles Osborne. 2006. Modelling lexical redundancy for machine translation. In *Proc. ACL*.
- Jörg Tiedemann. 2009. Character-based PSMT for closely related languages. In *Proc. 13th Annual Conference of the European Association for Machine Translation*.
- David Vilar, Jan-T. Peter, and Hermann Ney. 2007. Can we translate letters. In *Proc. WMT*.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proc. COLING*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3).
- Hao Zhang and Daniel Gildea. 2005. Stochastic lexicalized inversion transduction grammar for alignment. In *Proc. ACL*.
- Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008a. Bayesian learning of non-compositional phrases with synchronous parsing. *Proc. ACL*.
- Ruiqiang Zhang, Keiji Yasuda, and Eiichiro Sumita. 2008b. Improved statistical machine translation by multiple Chinese word segmentation. In *Proc. WMT*.

# Fast Syntactic Analysis for Statistical Language Modeling via Substructure Sharing and Uptraining

Ariya Rastrow, Mark Dredze, Sanjeev Khudanpur

Human Language Technology Center of Excellence

Center for Language and Speech Processing, Johns Hopkins University

Baltimore, MD USA

{ariya, mdredze, khudanpur}@jhu.edu

## Abstract

Long-span features, such as syntax, can improve language models for tasks such as speech recognition and machine translation. However, these language models can be difficult to use in practice because of the time required to generate features for rescoring a large hypothesis set. In this work, we propose substructure sharing, which saves duplicate work in processing hypothesis sets with redundant hypothesis structures. We apply substructure sharing to a dependency parser and part of speech tagger to obtain significant speedups, and further improve the accuracy of these tools through up-training. When using these improved tools in a language model for speech recognition, we obtain significant speed improvements with both  $N$ -best and hill climbing rescoring, and show that up-training leads to WER reduction.

## 1 Introduction

Language models (LM) are crucial components in tasks that require the generation of coherent natural language text, such as automatic speech recognition (ASR) and machine translation (MT). While traditional LMs use word  $n$ -grams, where the  $n - 1$  previous words predict the next word, newer models integrate long-span information in making decisions. For example, incorporating long-distance dependencies and syntactic structure can help the LM better predict words by complementing the predictive power of  $n$ -grams (Chelba and Jelinek, 2000; Collins et al., 2005; Filimonov and Harper, 2009; Kuo et al., 2009).

The long-distance dependencies can be modeled in either a *generative* or a *discriminative* framework. Discriminative models, which directly distinguish correct from incorrect hypothesis, are particularly attractive because they allow the inclusion of arbitrary features (Kuo et al., 2002; Roark et al., 2007; Collins et al., 2005); these models with syntactic information have obtained state of the art results.

However, both generative and discriminative LMs with long-span dependencies can be slow, for they often cannot work directly with lattices and require *rescoring* large  $N$ -best lists (Khudanpur and Wu, 2000; Collins et al., 2005; Kuo et al., 2009). For discriminative models, this limitation applies to training as well. Moreover, the non-local features used in rescoring are usually extracted via auxiliary tools – which in the case of syntactic features include part of speech taggers and parsers – from a set of ASR system hypotheses. Separately applying auxiliary tools to each  $N$ -best list hypothesis leads to major inefficiencies as many hypotheses differ only slightly.

Recent work on hill climbing algorithms for ASR lattice rescoring iteratively searches for a higher-scoring hypothesis in a local neighborhood of the current-best hypothesis, leading to a much more efficient algorithm in terms of the number,  $N$ , of hypotheses evaluated (Rastrow et al., 2011b); the idea also leads to a discriminative hill climbing training algorithm (Rastrow et al., 2011a). Even so, the reliance on auxiliary tools slow LM application to the point of being impractical for real time systems. While faster auxiliary tools are an option, they are usually less accurate.

In this paper, we propose a general modifica-

tion to the decoders used in auxiliary tools to utilize the commonalities among the set of generated hypotheses. The key idea is to share substructure states in transition based structured prediction algorithms, i.e. algorithms where final structures are composed of a sequence of multiple individual decisions. We demonstrate our approach on a local Perceptron based part of speech tagger (Tsuruoka et al., 2011) and a shift reduce dependency parser (Sagae and Tsujii, 2007), yielding significantly faster tagging and parsing of ASR hypotheses. While these simpler structured prediction models are faster, we compensate for the model’s simplicity through up-training (Petrov et al., 2010), yielding auxiliary tools that are both fast and accurate. The result is significant speed improvements and a reduction in word error rate (WER) for both  $N$ -best list and the already fast hill climbing rescoring. The net result is arguably the first syntactic LM fast enough to be used in a *real time ASR system*.

## 2 Syntactic Language Models

There have been several approaches to include syntactic information in both generative and discriminative language models.

For generative LMs, the syntactic information must be part of the generative process. Structured language modeling incorporates syntactic parse trees to identify the head words in a hypothesis for modeling dependencies beyond  $n$ -grams. Chelba and Jelinek (2000) extract the two previous exposed head words at each position in a hypothesis, along with their non-terminal tags, and use them as context for computing the probability of the current position. Khudanpur and Wu (2000) exploit such syntactic head word dependencies as features in a maximum entropy framework. Kuo et al. (2009) integrate syntactic features into a neural network LM for Arabic speech recognition.

Discriminative models are more flexible since they can include arbitrary features, allowing for a wider range of long-span syntactic dependencies. Additionally, discriminative models are directly trained to resolve the acoustic confusion in the decoded hypotheses of an ASR system. This flexibility and training regime translate into better performance. Collins et al. (2005) uses the Perceptron algorithm to train a global linear discriminative model

which incorporates long-span features, such as head-to-head dependencies and part of speech tags.

**Our Language Model.** We work with a discriminative LM with long-span dependencies. We use a global linear model with Perceptron training. We rescore the hypotheses (lattices) generated by the ASR decoder—in a framework most similar to that of Rastrow et al. (2011a).

The LM score  $\mathcal{S}(\mathbf{w}, \mathbf{a})$  for each hypothesis  $\mathbf{w}$  of a speech utterance with acoustic sequence  $\mathbf{a}$  is based on the baseline ASR system score  $b(\mathbf{w}, \mathbf{a})$  (initial  $n$ -gram LM score and the acoustic score) and  $\alpha_0$ , the weight assigned to the baseline score.<sup>1</sup> The score is defined as:

$$\begin{aligned} \mathcal{S}(\mathbf{w}, \mathbf{a}) &= \alpha_0 \cdot b(\mathbf{w}, \mathbf{a}) + F(\mathbf{w}, \mathbf{s}^1, \dots, \mathbf{s}^m) \\ &= \alpha_0 \cdot b(\mathbf{w}, \mathbf{a}) + \sum_{i=1}^d \alpha_i \cdot \Phi_i(\mathbf{w}, \mathbf{s}^1, \dots, \mathbf{s}^m) \end{aligned}$$

where  $F$  is the discriminative LM’s score for the hypothesis  $\mathbf{w}$ , and  $\mathbf{s}^1, \dots, \mathbf{s}^m$  are candidate syntactic structures associated with  $\mathbf{w}$ , as discussed below. Since we use a linear model, the score is a weighted linear combination of the *count* of activated features of the word sequence  $\mathbf{w}$  and its associated structures:  $\Phi_i(\mathbf{w}, \mathbf{s}^1, \dots, \mathbf{s}^m)$ . Perceptron training learns the parameters  $\alpha$ . The baseline score  $b(\mathbf{w}, \mathbf{a})$  can be a feature, yielding the dot product notation:  $\mathcal{S}(\mathbf{w}, \mathbf{a}) = \langle \alpha, \Phi(\mathbf{a}, \mathbf{w}, \mathbf{s}^1, \dots, \mathbf{s}^m) \rangle$  Our LM uses features from the dependency tree and part of speech (POS) tag sequence. We use the method described in Kuo et al. (2009) to identify the two previous exposed head words,  $h_{-2}, h_{-1}$ , at each position  $i$  in the input hypothesis and include the following syntactic based features into our LM:

1.  $(h_{-2}.w \circ h_{-1}.w \circ w_i), (h_{-1}.w \circ w_i), (w_i)$
2.  $(h_{-2}.t \circ h_{-1}.t \circ t_i), (h_{-1}.t \circ t_i), (t_i), (t_i w_i)$

where  $h.w$  and  $h.t$  denote the word identity and the POS tag of the corresponding exposed head word.

### 2.1 Hill Climbing Rescoring

We adopt the so called hill climbing framework of Rastrow et al. (2011b) to improve both training and rescoring time as much as possible by reducing the

<sup>1</sup>We tune  $\alpha_0$  on development data (Collins et al., 2005).

number  $N$  of explored hypotheses. We summarize it below for completeness.

Given a speech utterance’s lattice  $\mathcal{L}$  from a first pass ASR decoder, the neighborhood  $\mathcal{N}(\mathbf{w}, i)$  of a hypothesis  $\mathbf{w} = w_1 w_2 \dots w_n$  at position  $i$  is defined as the set of all paths in the lattice that may be obtained by editing  $w_i$ : deleting it, substituting it, or inserting a word to its left. In other words, it is the “distance-1-at-position  $i$ ” neighborhood of  $\mathbf{w}$ . Given a position  $i$  in a word sequence  $\mathbf{w}$ , all hypotheses in  $\mathcal{N}(\mathbf{w}, i)$  are rescored using the long-span model and the hypothesis  $\hat{\mathbf{w}}'(i)$  with the highest score becomes the new  $\mathbf{w}$ . The process is repeated with a new position – scanned left to right – until  $\mathbf{w} = \hat{\mathbf{w}}'(1) = \dots = \hat{\mathbf{w}}'(n)$ , i.e. when  $\mathbf{w}$  itself is the highest scoring hypothesis in all its 1-neighborhoods, and can not be further improved using the model. Incorporating this into training yields a discriminative hill climbing algorithm (Rastrow et al., 2011a).

### 3 Incorporating Syntactic Structures

Long-span models – generative or discriminative,  $N$ -best or hill climbing – rely on auxiliary tools, such as a POS tagger or a parser, for extracting features for each hypothesis during rescoring, and during training for discriminative models. The top- $m$  candidate structures associated with the  $i^{\text{th}}$  hypothesis, which we denote as  $\mathbf{s}_i^1, \dots, \mathbf{s}_i^m$ , are generated by these tools and used to score the hypothesis:  $F(\mathbf{w}_i, \mathbf{s}_i^1, \dots, \mathbf{s}_i^m)$ . For example,  $\mathbf{s}_i^j$  can be a part of speech tag or a syntactic dependency. We formally define this sequential processing as:

$$\begin{aligned} \mathbf{w}_1 &\xrightarrow{\text{tool}(s)} \mathbf{s}_1^1, \dots, \mathbf{s}_1^m \xrightarrow{\text{LM}} F(\mathbf{w}_1, \mathbf{s}_1^1, \dots, \mathbf{s}_1^m) \\ \mathbf{w}_2 &\xrightarrow{\text{tool}(s)} \mathbf{s}_2^1, \dots, \mathbf{s}_2^m \xrightarrow{\text{LM}} F(\mathbf{w}_2, \mathbf{s}_2^1, \dots, \mathbf{s}_2^m) \\ &\vdots \\ \mathbf{w}_k &\xrightarrow{\text{tool}(s)} \mathbf{s}_k^1, \dots, \mathbf{s}_k^m \xrightarrow{\text{LM}} F(\mathbf{w}_k, \mathbf{s}_k^1, \dots, \mathbf{s}_k^m) \end{aligned}$$

Here,  $\{\mathbf{w}_1, \dots, \mathbf{w}_k\}$  represents a set of ASR output hypotheses that need to be rescored. For each hypothesis, we apply an external tool (e.g. parser) to generate associated structures  $\mathbf{s}_i^1, \dots, \mathbf{s}_i^m$  (e.g. dependencies.) These are then passed to the language model along with the word sequence for scoring.

### 3.1 Substructure Sharing

While long-span LMs have been empirically shown to improve WER over  $n$ -gram LMs, the computational burden prohibits long-span LMs in practice, particularly in real-time systems. A major complexity factor is due to processing 100s or 1000s of hypotheses for each speech utterance, even during hill climbing, each of which must be POS tagged and parsed. However, the candidate hypotheses of an utterance share equivalent substructures, especially in hill climbing methods due to the locality present in the neighborhood generation. Figure 1 demonstrates such repetition in an  $N$ -best list ( $N=10$ ) and a hill climbing neighborhood hypothesis set for a speech utterance from broadcast news. For example, the word “ENDORSE” occurs within the same local context in all hypotheses and should receive the same part of speech tag in each case. Processing each hypothesis separately wastes time.

We propose a general algorithmic approach to reduce the complexity of processing a hypothesis set by sharing common substructures among the hypotheses. Critically, unlike many lattice parsing algorithms, our approach is general and produces exact output. We first present our approach and then demonstrate its generality by applying it to a dependency parser and part of speech tagger.

We work with structured prediction models that produce output from a series of local decisions: a transition model. We begin in initial state  $\pi_0$  and terminate in a possible final state  $\pi_f$ . All states along the way are chosen from the possible states  $\Pi$ . A transition (or action)  $\omega \in \Omega$  advances the decoder from state to state, where the transition  $\omega_i$  changes the state from  $\pi_i$  to  $\pi_{i+1}$ . The sequence of states  $\{\pi_0 \dots \pi_i, \pi_{i+1} \dots \pi_f\}$  can be mapped to an output (the model’s prediction.) The choice of action  $\omega$  is given by a learning algorithm, such as a maximum-entropy classifier, support vector machine or Perceptron, trained on labeled data. Given the previous  $k$  actions up to  $\pi_i$ , the classifier  $g : \Pi \times \Omega^k \rightarrow \mathbb{R}^{|\Omega|}$  assigns a score to each possible action, which we can interpret as a probability:  $p_g(\omega_i | \pi_i, \omega_{i-1} \omega_{i-2} \dots \omega_{i-k})$ . These actions are applied to transition to new states  $\pi_{i+1}$ . We note that state definitions can encode the  $k$  previous actions, which simplifies the probability to  $p_g(\omega_i | \pi_i)$ . The

<i>N</i> -best list	Hill climbing neighborhood
(1) AL GORE HAS PROMISED THAT HE WOULD ENDORSE A CANDIDATE	
(2) TO AL GORE HAS PROMISED THAT HE WOULD ENDORSE A CANDIDATE	
(3) AL GORE HAS PROMISE THAT HE WOULD ENDORSE A CANDIDATE	
(4) SO AL GORE HAS PROMISED THAT HE WOULD ENDORSE A CANDIDATE	(1) YEAH FIFTY CENT GALLON NOMINATION WHICH WAS GREAT
(5) IT'S AL GORE HAS PROMISED THAT HE WOULD ENDORSE A CANDIDATE	(2) YEAH FIFTY CENT A GALLON NOMINATION WHICH WAS GREAT
(6) AL GORE HAS PROMISED HE WOULD ENDORSE A CANDIDATE	(3) YEAH FIFTY CENT GOT A NOMINATION WHICH WAS GREAT
(7) AL GORE HAS PROMISED THAT HE WOULD ENDORSE THE CANDIDATE	
(8) SAID AL GORE HAS PROMISED THAT HE WOULD ENDORSE A CANDIDATE	
(9) AL GORE HAS PROMISED THAT HE WOULD ENDORSE A CANDIDATE FOR	
(10) AL GORE HIS PROMISE THAT HE WOULD ENDORSE A CANDIDATE	

Figure 1: Example of repeated substructures in candidate hypotheses.

score of the new state is then

$$p(\pi_{i+1}) = p_g(\omega_i | \pi_i) \cdot p(\pi_i) \quad (1)$$

Classification decisions require a feature representation of  $\pi_i$ , which is provided by feature functions  $\mathbf{f} : \Pi \rightarrow \mathcal{Y}$ , that map states to features. Features are conjoined with actions for multi-class classification, so  $p_g(\omega_i | \pi_i) = p_g(\mathbf{f}(\pi) \circ \omega_i)$ , where  $\circ$  is a conjunction operation. In this way, states can be summarized by features.

*Equivalent states* are defined as two states  $\pi$  and  $\pi'$  with an identical feature representation:

$$\pi \equiv \pi' \quad \text{iff} \quad \mathbf{f}(\pi) = \mathbf{f}(\pi')$$

If two states are equivalent, then  $g$  imposes the same distribution over actions. We can benefit from this substructure redundancy, both within and between hypotheses, by saving these distributions in memory, sharing a distribution computed just once across equivalent states. A similar idea of equivalent states is used by Huang and Sagae (2010), except they use equivalence to facilitate dynamic programming for shift-reduce parsing, whereas we generalize it for improving the processing time of similar hypotheses in general models. Following Huang and Sagae, we define *kernel features* as the smallest set of atomic features  $\tilde{\mathbf{f}}(\pi)$  such that,

$$\tilde{\mathbf{f}}(\pi) = \tilde{\mathbf{f}}(\pi') \quad \Rightarrow \quad \pi \equiv \pi'. \quad (2)$$

Equivalent distributions are stored in a *hash table*  $H : \Pi \rightarrow \Omega \times \mathbb{R}$ ; the hash keys are the states and the values are distributions<sup>2</sup> over actions:  $\{\omega, p_g(\omega | \pi)\}$ .

<sup>2</sup>For pure greedy search (deterministic search) we need only retain the best action, since the distribution is only used in probabilistic search, such as beam search or best-first algorithms.

$H$  caches equivalent states in a hypothesis set and re-sets for each new utterance. For each state, we first check  $H$  for equivalent states before computing the action distribution; each cache hit reduces decoding time. Distributing hypotheses  $w_i$  across different CPU threads is another way to obtain speedups, and we can still benefit from substructure sharing by storing  $H$  in shared memory.

We use  $h(\pi) = \sum_{i=1}^{|\tilde{\mathbf{f}}(\pi)|} \text{int}(f_i(\pi))$  as the hash function, where  $\text{int}(f_i(\pi))$  is an integer mapping of the  $i^{\text{th}}$  kernel feature. For integer typed features the mapping is trivial, for string typed features (e.g. a POS tag identity) we use a mapping of the corresponding vocabulary to integers. We empirically found that this hash function is very effective and yielded very few collisions.

To apply substructure sharing to a transition based model, we need only define the set of states  $\Pi$  (including  $\pi_0$  and  $\pi_f$ ), actions  $\Omega$  and kernel feature functions  $\tilde{\mathbf{f}}$ . The resulting speedup depends on the amount of substructure duplication among the hypotheses, which we will show is significant for ASR lattice rescoring. Note that our algorithm is not an approximation; we obtain the same output  $\{s_i^j\}$  as we would without any sharing. We now apply this algorithm to dependency parsing and POS tagging.

### 3.2 Dependency Parsing

We use the best-first probabilistic shift-reduce dependency parser of Sagae and Tsujii (2007), a transition-based parser (Kübler et al., 2009) with a MaxEnt classifier. Dependency trees are built by processing the words left-to-right and the classifier assigns a distribution over the actions at each step. States are defined as  $\pi = \{S, Q\}$ :  $S$  is a stack of

Kernel features $\mathbf{f}(\pi)$ for state $\pi = \{S, Q\}$ $S = s_0, s_1, \dots$ & $Q = q_0, q_1, \dots$				
(1)	$s_0.w$	$s_0.t$	$s_0.r$	(5) $t_{s_0-1}$ $t_{s_1+1}$
		$s_0.lch.t$	$s_0.lch.r$	
		$s_0.rch.t$	$s_0.rch.r$	
(2)	$s_1.w$	$s_1.t$	$s_1.r$	(6) $\text{dist}(s_0, s_1)$ $\text{dist}(q_0, s_0)$
		$s_1.lch.t$	$s_1.lch.r$	
		$s_1.rch.t$	$s_1.rch.r$	
(3)	$s_2.w$	$s_2.t$	$s_2.r$	
(4)	$q_0.w$	$q_0.t$		(7) $s_0.nch$ $s_1.nch$
	$q_1.w$	$q_1.t$		
	$q_2.w$			

Table 1: Kernel features for defining parser states.  $s_i.w$  denotes the head-word in a subtree and  $t$  its POS tag.  $s_i.lch$  and  $s_i.rch$  are the leftmost and rightmost children of a subtree.  $s_i.r$  is the dependency label that relates a subtree head-word to its dependent.  $s_i.nch$  is the number of children of a subtree.  $q_i.w$  and  $q_i.t$  are the word and its POS tag in the queue.  $\text{dist}(s_0, s_1)$  is the linear distance between the head-words of  $s_0$  and  $s_1$ .

subtrees  $s_0, s_1, \dots$  ( $s_0$  is the top tree) and  $Q$  are words in the input word sequence. The initial state is  $\pi_0 = \{\emptyset, \{w_0, w_1, \dots\}\}$ , and final states occur when  $Q$  is empty and  $S$  contains a single tree (the output).

$\Omega$  is determined by the set of dependency labels  $r \in \mathcal{R}$  and one of three transition types:

- *Shift*: remove the head of  $Q$  ( $w_j$ ) and place it on the top of  $S$  as a singleton tree (only  $w_j$ .)
- *Reduce-Left<sub>r</sub>*: replace the top two trees in  $S$  ( $s_0$  and  $s_1$ ) with a tree formed by making the root of  $s_1$  a dependent of the root of  $s_0$  with label  $r$ .
- *Reduce-Right<sub>r</sub>*: same as *Reduce-Left<sub>r</sub>* except reverses  $s_0$  and  $s_1$ .

Table 1 shows the kernel features used in our dependency parser. See Sagae and Tsujii (2007) for a complete list of features.

Goldberg and Elhadad (2010) observed that parsing time is dominated by feature extraction and score calculation. Substructure sharing reduces these steps for equivalent states, which are persistent throughout a candidate set. Note that there are far fewer kernel features than total features, hence the hash function calculation is very fast.

We summarize substructure sharing for dependency parsing in Algorithm 1. We extend the definition of states to be  $\{S, Q, p\}$  where  $p$  denotes the score of the state: the probability of the action sequence that resulted in the current state. Also, fol-

### Algorithm 1 Best-first shift-reduce dependency parsing

```

 $\mathbf{w} \leftarrow$  input hypothesis
 $S_0 = \emptyset, Q_0 = \mathbf{w}, p_0 = 1$ 
 $\pi_0 \leftarrow \{S_0, Q_0, p_0\}$  [initial state]
 $H \leftarrow$  Hash table ( $\Pi \rightarrow \Omega \times \mathbb{R}$ )
Heap  $\leftarrow$  Heap for prioritizing states and performing best-first search
Heap.push( $\pi_0$ ) [initialize the heap]

while Heap  $\neq \emptyset$  do
   $\pi_{\text{current}} \leftarrow$  Heap.pop() [the best state so far]
  if  $\pi_{\text{current}} = \pi_f$  [if final state]
    return  $\pi_{\text{current}}$  [terminate if final state]
  else if  $H.find(\pi_{\text{current}})$ 
    ActList  $\leftarrow H[\pi_{\text{current}}]$  [retrieve action list from the hash table]
  else [need to construct action list]
    for all  $\omega \in \Omega$  [for all actions]
       $p_\omega \leftarrow p_g(\omega | \pi_{\text{current}})$  [action score]
      ActList.insert( $\{\omega, p_\omega\}$ )
      H.insert( $\pi_{\text{current}}, \text{ActList}$ ) [Store the action list into hash table]
    end if
  for all  $\{\omega, p_\omega\} \in \text{ActList}$  [compute new states]
     $\pi_{\text{new}} \leftarrow \pi_{\text{current}} \times \omega$ 
    Heap.push( $\pi_{\text{new}}$ ) [push to the heap]
  end while

```

lowing Sagae and Tsujii (2007) a heap is used to maintain states prioritized by their scores, for applying the best-first strategy. For each step, a state from the top of the heap is considered and all actions (and scores) are either retrieved from  $H$  or computed using  $g$ .<sup>3</sup> We use  $\pi_{\text{new}} \leftarrow \pi_{\text{current}} \times \omega$  to denote the operation of extending a state by an action  $\omega \in \Omega^4$ .

### 3.3 Part of Speech Tagging

We use the part of speech (POS) tagger of Tsuruoka et al. (2011), a transition based model with a Perceptron and a lookahead heuristic process. The tagger processes  $\mathbf{w}$  left to right. States are defined as  $\pi_i = \{c_i, \mathbf{w}\}$ : a sequence of assigned tags up to  $w_i$  ( $c_i = t_1 t_2 \dots t_{i-1}$ ) and the word sequence  $\mathbf{w}$ .  $\Omega$  is defined simply as the set of possible POS tags ( $\mathcal{T}$ ) that can be applied. The final state is reached once all the positions are tagged. For  $\mathbf{f}$  we use the features of Tsuruoka et al. (2011). The kernel features are  $\mathbf{f}(\pi_i) = \{t_{i-2}, t_{i-1}, w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}\}$ . While the tagger extracts prefix and suffix features, it suffices to look at  $w_i$  for determining state equivalence. The tagger is deterministic (greedy) in that it only considers the best tag at each step, so we do not store scores. However, this tagger uses a depth-

<sup>3</sup> Sagae and Tsujii (2007) use a beam strategy to increase speed. Search space pruning is achieved by filtering heap states for probability greater than  $\frac{1}{b}$  the probability of the most likely state in the heap with the same number of actions. We use  $b = 100$  for our experiments.

<sup>4</sup>We note that while we have demonstrated substructure sharing for **dependency** parsing, the same improvements can be made to a shift-reduce constituent parser (Sagae and Lavie, 2006).

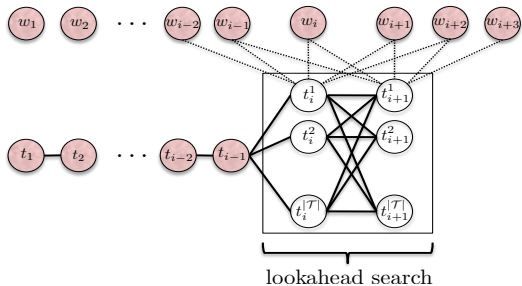


Figure 2: POS tagger with lookahead search of  $d=1$ . At  $w_i$  the search considers the current state and next state.

first search lookahead procedure to select the best action at each step, which considers future decisions up to depth  $d^5$ . An example for  $d = 1$  is shown in Figure 2. Using  $d = 1$  for the lookahead search strategy, we modify the kernel features since the decision for  $w_i$  is affected by the state  $\pi_{i+1}$ . The kernel features in position  $i$  should be  $\tilde{\mathbf{f}}(\pi_i) \cup \tilde{\mathbf{f}}(\pi_{i+1})$ :

$$\tilde{\mathbf{f}}(\pi_i) = \{t_{i-2}, t_{i-1}, w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}, w_{i+3}\}$$

## 4 Up-Training

While we have fast decoding algorithms for the parsing and tagging, the simpler underlying models can lead to worse performance. Using more complex models with higher accuracy is impractical because they are slow. Instead, we seek to improve the accuracy of our fast tools.

To achieve this goal we use up-training, in which a more complex model is used to improve the accuracy of a simpler model. We are given two models,  $M_1$  and  $M_2$ , as well as a large collection of unlabeled text. Model  $M_1$  is slow but very accurate while  $M_2$  is fast but obtains lower accuracy. Up-training applies  $M_1$  to tag the unlabeled data, which is then used as training data for  $M_2$ . Like self-training, a model is retrained on automatic output, but here the output comes from a more accurate model. Petrov et al. (2010) used up-training as a domain adaptation technique: a constituent parser – which is more robust to domain changes – was used to label a new domain, and a fast dependency parser

<sup>5</sup> Tsuruoka et al. (2011) shows that the lookahead search improves the performance of the local “history-based” models for different NLP tasks

was trained on the automatically labeled data. We use a similar idea where our goal is to recover the accuracy lost from using simpler models. Note that while up-training uses two models, it differs from co-training since we care about improving only one model ( $M_2$ ). Additionally, the models can vary in different ways. For example, they could be the same algorithm with different pruning methods, which can lead to faster but less accurate models.

We apply up-training to improve the accuracy of both our fast POS tagger and dependency parser. We parse a large corpus of text with a very accurate but very slow constituent parser and use the resulting data to up-train our tools. We will demonstrate empirically that up-training improves these fast models to yield better WER results.

## 5 Related Work

The idea of efficiently processing a hypothesis set is similar to “lattice-parsing”, in which a parser consider an entire lattice at once (Hall, 2005; Chempalier et al., 1999). These methods typically constrain the parsing space using heuristics, which are often model specific. In other words, they search in the joint space of word sequences present in the lattice and their syntactic analyses; they are not guaranteed to produce a syntactic analysis for all hypotheses. In contrast, substructure sharing is a general purpose method that we have applied to two different algorithms. The output is identical to processing each hypothesis separately and output is generated for each hypothesis. Hall (Hall, 2005) uses a lattice parsing strategy which aims to compute the marginal probabilities of all word sequences in the lattice by summing over syntactic analyses of each word sequence. The parser sums over multiple parses of a word sequence implicitly. The lattice parser therefore, is itself a language model. In contrast, our tools are completely separated from the ASR system, which allows the system to create whatever features are needed. This independence means our tools are useful for other tasks, such as machine translation. These differences make substructure sharing a more attractive option for efficient algorithms.

While Huang and Sagae (2010) use the notion of “equivalent states”, they do so for dynamic programming in a shift-reduce parser to broaden the search space. In contrast, we use the idea to identify sub-



structures across inputs, where our goal is efficient parsing in general. Additionally, we extend the definition of equivalent states to general transition based structured prediction models, and demonstrate applications beyond parsing as well as the novel setting of hypothesis set parsing.

## 6 Experiments

Our ASR system is based on the 2007 IBM Speech transcription system for the GALE Distillation Go/No-go Evaluation (Chen et al., 2006) with state of the art discriminative acoustic models. See Table 2 for a data summary. We use a modified Kneser-Ney (KN) backoff 4-gram baseline LM. Word-lattices for discriminative training and rescoring come from this baseline ASR system.<sup>6</sup> The long-span discriminative LM’s baseline feature weight ( $\alpha_0$ ) is tuned on dev data and hill climbing (Rastrow et al., 2011a) is used for training and rescoring. The dependency parser and POS tagger are trained on supervised data and up-trained on data labeled by the CKY-style bottom-up constituent parser of Huang et al. (2010), a state of the art broadcast news (BN) parser, with phrase structures converted to labeled dependencies by the Stanford converter.

While accurate, the parser has a huge grammar (32GB) from using products of latent variable grammars and requires  $O(l^3)$  time to parse a sentence of length  $l$ . Therefore, we could not use the constituent parser for ASR rescoring since utterances can be very long, although the shorter up-training text data was not a problem.<sup>7</sup> We evaluate both unlabeled (UAS) and labeled dependency accuracy (LAS).

### 6.1 Results

Before we demonstrate the speed of our models, we show that up-training can produce accurate and fast models. Figure 3 shows improvements to parser accuracy through up-training for different amount of (randomly selected) data, where the last column indicates constituent parser score (91.4% UAS). We use the POS tagger to generate tags for dependency training to match the test setting. While there is a large difference between the constituent and dependency parser without up-training (91.4%

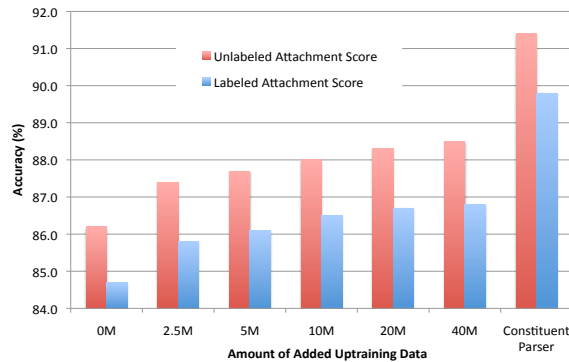


Figure 3: Up-training results for dependency parsing for varying amounts of data (number of words.) The first column is the dependency parser with supervised training only and the last column is the constituent parser (after converting to dependency trees.)

vs. 86.2% UAS), up-training can cut the difference by 44% to 88.5%, and improvements saturate around 40m words (about 2m sentences.)<sup>8</sup> The dependency parser remains much smaller and faster; the up-trained dependency model is 700MB with 6m features compared with 32GB for constituency model. Up-training improves the POS tagger’s accuracy from 95.9% to 97%, when trained on the POS tags produced by the constituent parser, which has a tagging accuracy of 97.2% on BN.

We train the syntactic discriminative LM, with head-word and POS tag features, using the faster parser and tagger and then rescore the ASR hypotheses. Table 3 shows the decoding speedups as well as the WER reductions compared to the baseline LM. Note that up-training improvements lead to WER reductions. Detailed speedups on substructure sharing are shown in Table 4; the POS tagger achieves a 5.3 times speedup, and the parser a 5.7 speedup without changing the output. We also observed speedups during training (not shown due to space.)

The above results are for the already fast hill climbing decoding, but substructure sharing can also be used for  $N$ -best list rescoring. Figure 4 (logarithmic scale) illustrates the time for the parser and tagger to process  $N$ -best lists of varying size, with more substantial speedups for larger lists. For example, for  $N=100$  (a typical setting) the parsing time re-

<sup>6</sup>For training a 3-gram LM is used to increase confusions.

<sup>7</sup>Speech utterances are longer as they are not as effectively sentence segmented as text.

<sup>8</sup>Better performance is due to the exact CKY-style – compared with best-first and beam-search and that the constituent parser uses the product of huge self-trained grammars.

Usage	Data	Size
Acoustic model training	Hub4 acoustic train	153k uttr, 400 hrs
Baseline LM training: modified KN 4-gram	TDT4 closed captions+EARS BN03 closed caption	193m words
Disc. LM training: long-span w/hill climbing	Hub4 (length <50)	115k uttr, 2.6m words
Baseline feature ( $\alpha_0$ ) tuning	dev04f BN data	2.5 hrs
Supervised training: dep. parser, POS tagger	Ontonotes BN treebank+ WSJ Penn treebank	1.3m words, 59k sent.
Supervised training: constituent parser	Ontonotes BN treebank + WSJ Penn treebank	1.3m words, 59k sent.
Up-training: dependency parser, POS tagger	TDT4 closed captions+EARS BN03 closed caption	193m words available
Evaluation: up-training	BN treebank test (following Huang et al. (2010))	20k words, 1.1k sent.
Evaluation: ASR transcription	rt04 BN evaluation	4 hrs, 45k words

Table 2: A summary of the data for training and evaluation. The Ontonotes corpus is from Weischedel et al. (2008).

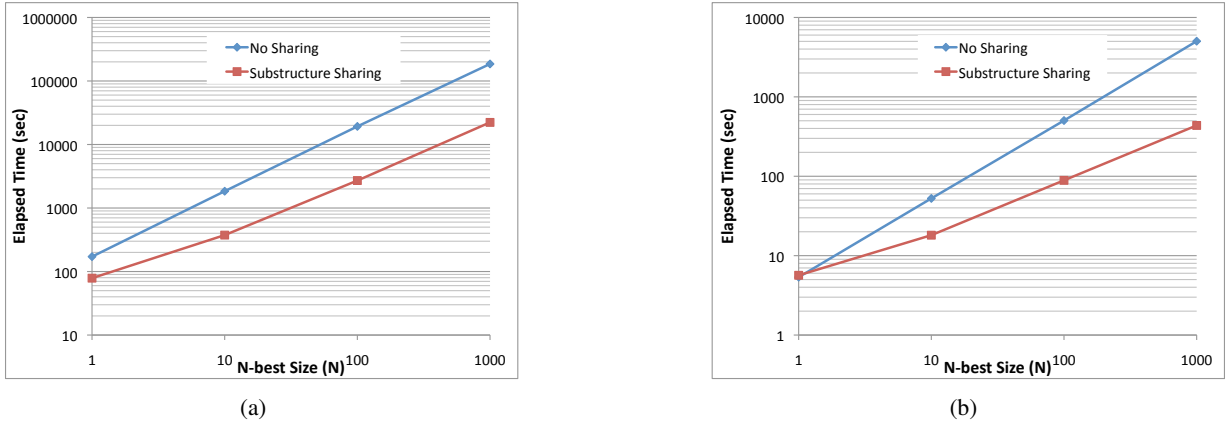


Figure 4: Elapsed time for (a) parsing and (b) POS tagging the  $N$ -best lists with and without substructure sharing.

LM	WER	Substr. Share (sec)	
		No	Yes
Baseline 4-gram	15.1	-	-
Syntactic LM + up-train	14.8 <b>14.6</b>	8,658	<b>1,648</b>

Table 3: Speedups and WER for hill climbing rescoring. Substructure sharing yields a 5.3 times speedup. The times for with and without up-training are nearly identical, so we include only one set for clarity. Time spent is dominated by the parser, so the faster parser accounts for much of the overall speedup. Timing information includes neighborhood generation and LM rescoring, so it is more than the sum of the times in Table 4.

duces from about 20,000 seconds to 2,700 seconds, about 7.4 times as fast.

## 7 Conclusion

The computational complexity of accurate syntactic processing can make structured language models impractical for applications such as ASR that require scoring hundreds of hypotheses per input. We have

	Substr. Share		Speedup
	No	Yes	
Parser	8,237.2	<b>1,439.5</b>	5.7
POS tagger	213.3	<b>40.1</b>	5.3

Table 4: Time in seconds for the parser and POS tagger to process hypotheses during hill climbing rescoring.

presented substructure sharing, a general framework that greatly improves the speed of syntactic tools that process candidate hypotheses. Furthermore, we achieve improved performance through up-training. The result is a large speedup in rescoring time, even on top of the already fast hill climbing framework, and reductions in WER from up-training. Our results make long-span syntactic LMs practical for real-time ASR, and can potentially impact machine translation decoding as well.

## Acknowledgments

Thanks to Kenji Sagae for sharing his shift-reduce dependency parser and the anonymous reviewers for helpful comments.

## References

- C. Chelba and F. Jelinek. 2000. Structured language modeling. *Computer Speech and Language*, 14(4):283–332.
- S. Chen, B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau, and G. Zweig. 2006. Advances in speech transcription at IBM under the DARPA EARS program. *IEEE Transactions on Audio, Speech and Language Processing*, pages 1596–1608.
- J. Cheppalier, M. Rajman, R. Aragues, and A. Rozenknop. 1999. Lattice parsing for speech recognition. In *Sixth Conference sur le Traitement Automatique du Langage Naturel (TANL'99)*.
- M Collins, B Roark, and M Saraclar. 2005. Discriminative syntactic language modeling for speech recognition. In *ACL*.
- Denis Filimonov and Mary Harper. 2009. A joint language model with fine-grain syntactic tags. In *EMNLP*.
- Yoav Goldberg and Michael Elhadad. 2010. An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Proc. HLT-NAACL*, number June, pages 742–750.
- Keith B Hall. 2005. *Best-first word-lattice parsing: techniques for integrated syntactic language modeling*. Ph.D. thesis, Brown University.
- L. Huang and K. Sagae. 2010. Dynamic Programming for Linear-Time Incremental Parsing. In *Proceedings of ACL*.
- Zhongqiang Huang, Mary Harper, and Slav Petrov. 2010. Self-training with Products of Latent Variable Grammars. In *Proc. EMNLP*, number October, pages 12–22.
- S. Khudanpur and J. Wu. 2000. Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling. *Computer Speech and Language*, pages 355–372.
- S. Kübler, R. McDonald, and J. Nivre. 2009. Dependency parsing. *Synthesis Lectures on Human Language Technologies*, 2(1):1–127.
- Hong-Kwang Jeff Kuo, Eric Fosler-Lussier, Hui Jiang, and Chin-Hui Lee. 2002. Discriminative training of language models for speech recognition. In *ICASSP*.
- H. K. J. Kuo, L. Mangu, A. Emami, I. Zitouni, and L. Young-Suk. 2009. Syntactic features for Arabic speech recognition. In *Proc. ASRU*.
- Slav Petrov, Pi-Chuan Chang, Michael Ringgaard, and Hiyan Alshawi. 2010. Uptraining for accurate deterministic question parsing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 705–713, Cambridge, MA, October. Association for Computational Linguistics.
- Ariya Rastrow, Mark Dredze, and Sanjeev Khudanpur. 2011a. Efficient discriminative training of long-span language models. In *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.
- Ariya Rastrow, Markus Dreyer, Abhinav Sethy, Sanjeev Khudanpur, Bhuvana Ramabhadran, and Mark Dredze. 2011b. Hill climbing on speech lattices : A new rescoring framework. In *ICASSP*.
- Brian Roark, Murat Saraclar, and Michael Collins. 2007. Discriminative n-gram language modeling. *Computer Speech & Language*, 21(2).
- K. Sagae and A. Lavie. 2006. A best-first probabilistic shift-reduce parser. In *Proc. ACL*, pages 691–698. Association for Computational Linguistics.
- K. Sagae and J. Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proc. EMNLP-CoNLL*, volume 7, pages 1044–1050.
- Yoshimasa Tsuruoka, Yusuke Miyao, and Jun'ichi Kazama. 2011. Learning with Lookahead : Can History-Based Models Rival Globally Optimized Models ? In *Proc. CoNLL*, number June, pages 238–246.
- Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, and Ann Houston, 2008. *OntoNotes Release 2.0*. Linguistic Data Consortium, Philadelphia.

# Bootstrapping a Unified Model of Lexical and Phonetic Acquisition

**Micha Elsner**

melsner0@gmail.com  
ILCC, School of Informatics  
University of Edinburgh  
Edinburgh, EH8 9AB, UK

**Sharon Goldwater**

sgwater@inf.ed.ac.uk  
ILCC, School of Informatics  
University of Edinburgh  
Edinburgh, EH8 9AB, UK

**Jacob Eisenstein**

jacobe@gmail.com  
School of Interactive Computing  
Georgia Institute of Technology  
Atlanta, GA, 30308, USA

## Abstract

During early language acquisition, infants must learn both a lexicon and a model of phonetics that explains how lexical items can vary in pronunciation—for instance “the” might be realized as [ði] or [ðə]. Previous models of acquisition have generally tackled these problems in isolation, yet behavioral evidence suggests infants acquire lexical and phonetic knowledge simultaneously. We present a Bayesian model that clusters together phonetic variants of the same lexical item while learning both a language model over lexical items and a log-linear model of pronunciation variability based on articulatory features. The model is trained on transcribed surface pronunciations, and learns by bootstrapping, without access to the true lexicon. We test the model using a corpus of child-directed speech with realistic phonetic variation and either gold standard or automatically induced word boundaries. In both cases modeling variability improves the accuracy of the learned lexicon over a system that assumes each lexical item has a unique pronunciation.

## 1 Introduction

Infants acquiring their first language confront two difficult cognitive problems: building a lexicon of word forms, and learning basic phonetics and phonology. The two tasks are closely related: knowing what sounds can substitute for one another helps in clustering together variant pronunciations of the same word, while knowing the environments in which particular words can occur helps determine which sound changes are meaningful and which are not (Feldman

(a) intended:	/ju want wʌn/	/want e kʊki/
(b) surface:	[jə wãʔ wʌn]	[wan ə kʊki]
(c) unsegmented:	[jəwãʔwʌn]	[wanəkʊki]
(d) idealized:	/juwantwʌn/	/wantekʊki/

Figure 1: The utterances *you want one? want a cookie?* represented (a) using a canonical phonemic encoding for each word and (b) as they might be pronounced phonetically. Lines (c) and (d) remove the word boundaries (but not utterance boundaries) from (b) and (a), respectively.

et al., 2009). For instance, if an infant who already knows the word [ju] “you” encounters a new word [jə], they must decide whether it is a new lexical item or a variant of the word they already know. Evidence for the correct conclusion comes from the pronunciation (many English vowels are reduced to [ə] in unstressed positions) and the context—if the next word is “want”, “you” is a plausible choice.

To date, most models of infant language learning have focused on either lexicon-building or phonetic learning in isolation. For example, many models of word segmentation implicitly or explicitly build a lexicon while segmenting the input stream of phonemes into word tokens; in nearly all cases the phonemic input is created from an orthographic transcription using a phonemic dictionary, thus abstracting away from any phonetic variability (Brent, 1999; Venkataraman, 2001; Swingley, 2005; Goldwater et al., 2009, among others). As illustrated in Figure 1, these models attempt to infer line (a) from line (d). However, (d) is an idealization: real speech has variability, and behavioral evidence suggests that infants are still learning about the phonetics and phonology of their language even after beginning to segment words, rather than learning to neutralize

the variations first and acquiring the lexicon afterwards (Feldman et al., 2009, and references therein).

Based on this evidence, a more realistic model of early language acquisition should propose a method of inferring the *intended* forms (Figure 1a) from the *unsegmented surface* forms (1c) while also learning a model of phonetic variation relating the intended and surface forms (a) and (b). Previous models with similar goals have learned from an artificial corpus with a small vocabulary (Driesen et al., 2009; Räsänen, 2011) or have modeled variability only in vowels (Feldman et al., 2009); to our knowledge, this paper is the first to use a naturalistic infant-directed corpus while modeling variability in all segments, and to incorporate word-level context (a bigram language model). Our main contribution is a joint lexical-phonetic model that infers intended forms from *segmented* surface forms; we test the system using input with either gold standard word boundaries or boundaries induced by an existing unsupervised segmentation model (Goldwater et al., 2009). We show that in both cases modeling variability improves the accuracy of the learned lexicon over a system that assumes each intended form has a unique surface form.

Our model is conceptually similar to those used in speech recognition and other applications: we assume the intended tokens are generated from a bigram language model and then distorted by a noisy channel, in particular a log-linear model of phonetic variability. But unlike speech recognition, we have no (intended-form, surface-form) training pairs to train the phonetic model, nor even a dictionary of intended-form strings to train the language model. Instead, we initialize the noise model using feature weights based on universal linguistic principles (e.g., a surface phone is likely to share articulatory features with the intended phone) and use a bootstrapping process to iteratively infer the intended forms and retrain the language model and noise model. While we do not claim that the particular inference mechanism we use is cognitively plausible, our positive results further support the claim that infants can and do acquire phonetics and the lexicon in concert.

## 2 Related work

Our work is inspired by the lexical-phonetic model of Feldman et al. (2009). They extend a model for

clustering acoustic tokens into phonetic categories (Vallabha et al., 2007) by adding a lexical level that simultaneously clusters word tokens (which contain the acoustic tokens) into lexical entries. Including the lexical level improves the model’s phonetic categorization, and a follow-up study on artificial language learning (Feldman, 2011) supports the claim that human learners use lexical knowledge to distinguish meaningful from unimportant phonetic contrasts. Feldman et al. (2009) use a real-valued representation for vowels (formant values), but assume no variability in consonants, and treat each word token independently. In contrast, our model uses a symbolic representation for sounds, but models variability in *all* segment types and incorporates a bigram word-level language model.

To our knowledge, the only other lexicon-building systems that also learn about phonetic variability are those of Driesen et al. (2009) and Räsänen (2011). These systems learn to represent lexical items and their variability from a discretized representation of the speech stream, but they are tested on an artificial corpus with only 80 vocabulary items that was constructed so as to “avoid strong word-to-word dependencies” (Räsänen, 2011). Here, we use a naturalistic corpus, demonstrating that lexical-phonetic learning is possible in this more general setting and that word-level context information is important for doing so.

Several other related systems work directly from the acoustic signal and many of these do use naturalistic corpora. However, they do not learn at both the lexical and phonetic/acoustic level. For example, Park and Glass (2008), Aimetti (2009), Jansen et al. (2010), and McInnes and Goldwater (2011) present lexicon-building systems that use hard-coded acoustic similarity measures rather than *learning* about variability, and they only extract and cluster a few frequent words. On the phonetic side, Varadarajan et al. (2008) and Dupoux et al. (2011) describe systems that learn phone-like units but without the benefit of top-down information.

A final line of related work is on word segmentation. In addition to the models mentioned in Section 1, which use phonemic input, a few models of word segmentation have been tested using phonetic input (Fleck, 2008; Rytting, 2007; Daland and Pierrehumbert, 2010). However, they do not cluster segmented

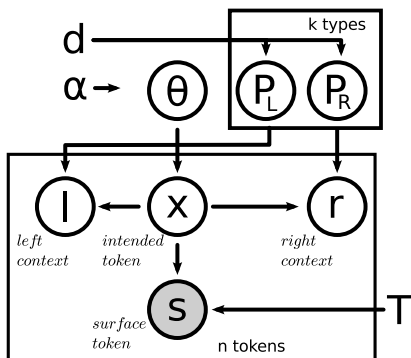


Figure 2: Our generative model of the surface tokens  $s$  from intended tokens  $x$ , which occur with left and right contexts  $l$  and  $r$ .

word tokens into lexical items (none of these models even maintains an explicit lexicon), nor do they model or learn from phonetic variation in the input.

### 3 Lexical-phonetic model

Our lexical-phonetic model is defined using the standard noisy channel framework: first a sequence of intended word tokens is generated using a language model, and then each token is transformed by a probabilistic finite-state transducer to produce the observed surface sequence. In this section, we present the model in a hierarchical Bayesian framework to emphasize its similarity to existing models, in particular those of Feldman et al. (2009) and Goldwater et al. (2009). In our actual implementation, however, we use approximation and MAP point estimates to make our inference process more tractable; we discuss these simplifications in Section 4.

Our observed data consists of a (segmented) sequence of surface words  $s_1 \dots s_n$ . We wish to recover the corresponding sequence of intended words  $x_1 \dots x_n$ . As shown in Figure 2,  $s_i$  is produced from  $x_i$  by a transducer  $T$ :  $s_i \sim T(x_i)$ , which models phonetic changes. Each  $x_i$  is sampled from a distribution  $\theta$  which represents word frequencies, and its left and right context words,  $l_i$  and  $r_i$ , are drawn from distributions conditioned on  $x_i$ , in order to capture information about the environments in which  $x_i$  appears:  $l_i \sim P_L(x_i), r_i \sim P_R(x_i)$ . Because the number of word types is not known in advance,  $\theta$  is drawn from a Dirichlet process  $DP(\alpha)$ , and  $P_L(x)$  and  $P_R(x)$  have Pitman-Yor priors with concentration parameter  $\theta$  and discount  $d$  (Teh, 2006).

Our generative model of  $x_i$  is unusual for two reasons. First, we treat each  $x_i$  independently rather than linking them via a Markov chain. This makes the model deficient, since  $l_i$  overlaps with  $x_{i-1}$  and so forth, generating each token twice. During inference, however, we will never compute the joint probability of all the data at once, only the probabilities of subsets of the variables with particular intended word forms  $u$  and  $v$ . As long as no two of these words are adjacent, the deficiency will have no effect. We make this independence assumption for computational reasons—when deciding whether to merge  $u$  and  $v$  into a single lexical entry, we compute the change in estimated probability for their contexts, but not the effect on other words for which  $u$  and  $v$  themselves appear as context words.

Also unusual is that we factor the joint probability  $(l, x, r)$  as  $p(x)p(l|x)p(r|x)$  rather than as a left-to-right chain  $p(l)p(x|l)p(r|x)$ . Given our independence assumption above, these two quantities are mathematically equivalent, so the difference matters only because we are using smoothed estimates. Our factorization leads to a symmetric treatment of left and right contexts, which simplifies implementation: we can store all the context parameters locally as  $P_L(\cdot|x)$  rather than distributed over various  $P(x|\cdot)$ .

Next, we explain our transducer  $T$ . A weighted finite-state transducer (WFST) is a variant of a finite-state automaton (Pereira et al., 1994) that reads an input string symbol-by-symbol and probabilistically produces an output string; thus it can be used to specify a conditional probability on output strings given an input. Our WFST (Figure 3) computes a weighted edit distance, and is implemented using OpenFST (Allauzen et al., 2007). It contains a state for each triplet of (*previous*, *current*, *next*) phones; conditioned on this state, it emits a character *output* which can be thought of as a possible surface realization of *current* in its particular environment. The *output* can be the empty string  $\epsilon$ , in which case *current* is deleted. The machine can also insert characters at any point in the string, by transitioning to an insert state (*previous*,  $\epsilon$ , *current*) and then returning while emitting some new character.

The transducer is parameterized by the probabilities of the arcs. For instance, all arcs leaving the state  $(\bullet, \delta, i)$  consume the character  $\delta$  and emit some character  $c$  with probability  $p(c|\bullet, \delta, i)$ . Following

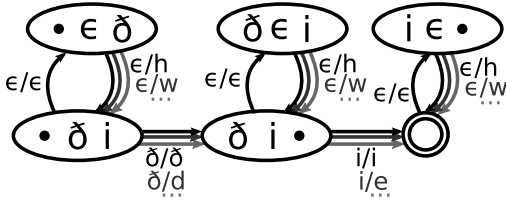


Figure 3: The fragment of the transducer responsible for input string [ǔi] “the”. “...” represents an output arc for each possible character, including the empty string  $\epsilon$ ;  $\bullet$  is the word boundary marker.

Dreyer et al. (2008), we parameterize these distributions with a log-linear model. The model features are based on articulatory phonetics and distinguish three dimensions of sound production: voicing, place of articulation and manner of articulation.

Features are generated from four positional templates (Figure 4):  $(curr) \rightarrow out$ ,  $(prev, curr) \rightarrow out$ ,  $(curr, next) \rightarrow out$  and  $(prev, curr, next) \rightarrow out$ . Each template is instantiated once per articulatory dimension, with  $prev$ ,  $curr$ ,  $next$  and  $out$  replaced by their values for that dimension: for instance, there are two voicing values, *voiced* and *unvoiced*<sup>1</sup> and the  $(curr) \rightarrow out$  template for [ǔ] producing [d] would be instantiated as  $(voiced) \rightarrow voiced$ . To capture trends specific to particular sounds, each template is instantiated again using the actual symbol for  $curr$  and articulatory values for everything else (e.g., [ǔ]  $\rightarrow$  *unvoiced*). An additional template,  $\rightarrow out$ , captures the marginal frequency of the output symbol. There are also faithfulness features, *same-sound*, *same-voice*, *same-place* and *same-manner* which check if  $curr$  is exactly identical to  $out$  or shares the exact value of a particular feature.

Our choice of templates and features is based on standard linguistic principles: we expect that changing only a single articulatory dimension will be more acceptable than changing several, and that the articulatory dimensions of context phones are important because of assimilatory and dissimilatory processes (Hayes, 2011). In modern phonetics and phonology, these generalizations are usually expressed as Optimality Theory constraints; log-linear models such as ours have previously been used to implement stochas-

<sup>1</sup>We use seven place values and five manner values (stop, nasal stop, fricative, vowel, other). Empty segments like  $\epsilon$  and  $\bullet$  are assigned a special value “no-value” for all features.

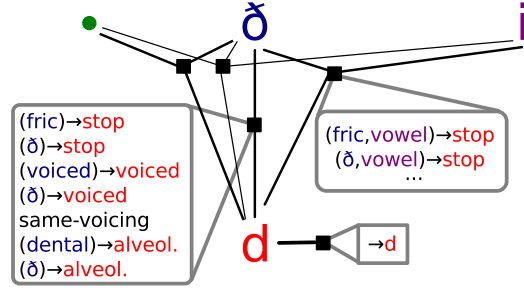


Figure 4: Some features generated for  $(\bullet, \ddot{u}, i) \rightarrow d$ . Each black factor node corresponds to a positional template. The features instantiated for the  $(curr) \rightarrow out$  and  $\rightarrow out$  template are shown in full, and we show some of the features for the  $(curr, next) \rightarrow out$  template.

tic Optimality Theory models (Goldwater and Johnson, 2003; Hayes and Wilson, 2008).

## 4 Inference

Global optimization of the model posterior is difficult; instead we use Viterbi EM (Spitkovsky et al., 2010; Allahverdyan and Galstyan, 2011). We begin with a simple initial transducer and alternate between two phases: clustering together surface forms, and reestimating the transducer parameters. We iterate this procedure until convergence (when successive clustering phases find nearly the same set of merges); this tends to take about 5 or 6 iterations.

In our clustering phase, we improve the model posterior as much as possible by greedily making *type merges*, where, for a pair of intended word forms  $u$  and  $v$ , we replace all instances of  $x_i = u$  with  $x_i = v$ . We maintain the invariant that each intended word form’s most common surface form must be itself; this biases the model toward solutions with low distortion in the transducer.

### 4.1 Scoring merges

We write the change in the log posterior probability of the model resulting from a type merge of  $u$  to  $v$  as  $\Delta(u, v)$ , which factors into two terms, one depending on the surface string and the transducer, and the other depending on the string of intended words. In order to ensure that each intended word form’s most common surface form is itself, we define  $\Delta(u, v) = -\infty$  if  $u$  is more common than  $v$ .

We write the log probability of  $x$  being transduced to  $s$  as  $T(s|x)$ . If we merge  $u$  into  $v$ , we no longer



need to produce any surface forms from  $u$ , but instead we must derive them from  $v$ . If  $\#(\cdot)$  counts the occurrences of some event in the current state of the model, the transducer component of  $\Delta$  is:

$$\Delta_T = \sum_s \#(x_i=u, s_i=s)(T(s|v) - T(s|u)) \quad (1)$$

This term is typically negative, voting against a merge, since  $u$  is more similar to itself than to  $v$ .

The language modeling term relating to the intended string again factors into multiple components. The probability of a particular  $l_i, x_i, r_i$  can be broken into  $p(x_i)p(l_i|x_i)p(r_i|x_i)$  according to the model. We deal first with the  $p(x_i)$  unigram term, considering all tokens where  $x_i \in \{u, v\}$  and computing the probability  $p_u = p(x_i = u|x_i \in \{u, v\})$ . By definition of a Dirichlet process, the marginal over a subset of the variables will be Dirichlet, so for  $\alpha > 1$  we have the MAP estimate:

$$p_u = \frac{\#(x_i=u) + \alpha - 1}{\#(x_i \in \{u, v\}) + 2(\alpha - 1)} \quad (2)$$

$p_v = p(x_i = v|x_i \in \{u, v\})$  is computed similarly. If we decide to merge  $u$  into  $v$ , however, the probability  $p(x_i = v|x_i \in \{u, v\})$  becomes 1. The change in log-probability resulting from the merge is closely related to the entropy of the distribution:

$$\Delta_U = -\#(x_i=u) \log(p_u) - \#(x_i=v) \log(p_v) \quad (3)$$

This change must be positive and favors merging.

Next, we consider the change in probability from the left contexts (the derivations for right contexts are equivalent). If  $u$  and  $v$  are separate words, we generate their left contexts from different distributions  $p(l|u)$  and  $p(l|v)$ , while if they are merged, we must generate all the contexts from the same distribution  $p(l|\{u, v\})$ . This change is:

$$\begin{aligned} \Delta_L = & \sum_l \#(l, u) \{ \log(p(l|\{u, v\})) - \log(p(l|u)) \} \\ & + \sum_l \#(l, v) \{ \log(p(l|\{u, v\})) - \log(p(l|v)) \} \end{aligned}$$

In a full Bayesian model, we would integrate over the parameters of these distributions; instead, we use Kneser-Ney smoothing (Kneser and Ney, 1995) which has been shown to approximate the MAP solution of a hierarchical Pitman-Yor model (Teh, 2006;

Goldwater et al., 2006). The Kneser-Ney discount<sup>2</sup>  $d$  is a tunable parameter of our system, and controls whether the term favors merging or not. If  $d$  is small,  $p(l|u)$  and  $p(l|v)$  are close to their maximum-likelihood estimates, and  $\Delta_L$  is similar to a Jensen-Shannon divergence; it is always negative and discourages mergers. As  $d$  increases, however,  $p(l|u)$  for rare words approaches the prior distribution; in this case, merging two words may result in better posterior parameters than estimating both separately, since the combined estimate loses less mass to discounting.

Because neither the transducer nor the language model are perfect models of the true distribution, they can have incompatible dynamic ranges. Often, the transducer distribution is too peaked; to remedy this, we downweight the transducer probability by  $\lambda$ , a parameter of our model, which we set to .5. Downweighting of the acoustic model versus the LM is typical in speech recognition (Bahl et al., 1980).

To summarize, the full change in posterior is:

$$\Delta(u, v) = \Delta_U + \Delta_L + \Delta_R + \lambda \Delta_T \quad (4)$$

There are four parameters. The transducer regularization  $r = 1$  and unigram prior  $\alpha = 2$ , which we set ad-hoc, have little impact on performance. The Kneser-Ney discount  $d = 2$  and transducer downweight  $\lambda = .5$  have more influence and were tuned on development data.

## 4.2 Clustering algorithm

In the clustering phase, we start with an initial solution in which each surface form is its own intended pronunciation and iteratively improve this solution by merging together word types, picking (approximately) the best merger at each point.

We begin by computing a set of *candidate* mergers for each surface word type  $u$ . This step saves time by quickly rejecting mergers which are certain to get very low transducer scores. We reject a pair  $u, v$  if the difference in their length is greater than 4, or if both words are longer than 4 segments, but, when we consider them as unordered bags of segments, the Dice coefficient between them is less than .5.

For each word  $u$  and all its candidates  $v$ , we compute  $\Delta(u, v)$  as in Equation 4. We keep track of the

<sup>2</sup>We use one discount, rather than several as in modified KN.



**Input:** vocabulary of surface forms  $u$   
**Input:**  $C(u)$ : candidate intended forms of  $u$   
**Output:**  $intend(u)$ : intended form of  $u$   
**foreach**  $u \in vocab$  **do**  
  // initialization  
   $v^*(u) \leftarrow \operatorname{argmax}_{v \in C(u)} \Delta(u, v)$ ;  
   $\Delta^*(u) \leftarrow \Delta(u, v^*(u))$   
   $intend(u) \leftarrow u$   
  add  $u$  to queue  $Q$  with priority  $\Delta^*(u)$   
**while**  $top(Q) > -\infty$  **do**  
   $u \leftarrow pop(Q)$   
  recompute  $v^*(u), \Delta^*(u)$   
  **if**  $\Delta^*(u) > 0$  **then**  
    // merge  $u$  with best merger  
     $intend(u) \leftarrow v^*(u)$   
    update  $\Delta(x, u) \quad \forall x : v^*(x) = u$   
    remove  $u$  from  $C(x) \quad \forall x$   
    update  $\Delta(x, v) \quad \forall x : v^*(x) = v$   
    update  $\Delta(v, x) \quad \forall x \in C(v)$   
    **if** updated  $\Delta > \Delta^*$  for any words **then**  
      | reset  $\Delta^*, v^*$  for those words  
    // (these updates can  
      increase a word's priority  
      from  $-\infty$ )  
  **else if**  $\Delta^*(u) \neq -\infty$  **then**  
    // reject but leave in queue  
     $\Delta^*(u) \leftarrow -\infty$

**Algorithm 1:** Our clustering phase.

current best target  $v^*(u)$  and best score  $\Delta^*(u)$ , using a priority queue. At each step of the algorithm, we pop the  $u$  with the current best  $\Delta^*(u)$ , recompute its scores, and then merge it with  $v^*(u)$  if doing so would improve the model posterior. In an exact algorithm, we would then need to recompute most of the other scores, since merging  $u$  and  $v^*(u)$  affects other words for which  $u$  and  $v^*(u)$  are candidates, and also words for which they appear in the context set. However, recomputing all these scores would be extremely time-consuming.<sup>3</sup> Therefore, we recompute scores for only those words where the previous best merger was either  $u$  or  $v^*(u)$ . (If the best merge would *not* improve the probability, we reject it, but since its score might increase if we merge  $v^*(u)$ , we leave  $u$  in the queue, setting its  $\Delta$  score to  $-\infty$ ; this score will be updated if we merge  $v^*(u)$ .)

Since we recompute the exact scores  $\Delta(u, v)$  immediately before merging  $u$ , the algorithm is guaran-

<sup>3</sup>The transducer scores can be cached since they depend only on surface forms, but the language model scores cannot.

teed never to reduce the posterior probability. It can potentially make changes in the wrong order, since not all the  $\Delta$ s are recomputed in each step, but most changes do not affect one another, so performing them out of order has no impact. Empirically, we find that mutually exclusive changes (usually of the form  $(u, v)$  and  $(v, w)$ ) tend to differ enough in initial score that they are evaluated in the correct order.

### 4.3 Training the transducer

To train the transducer on a set of mappings between surface and intended forms, we find the maximum-probability state sequence for each mapping (another application of Viterbi EM) and extract features for each state and its output. Learning weights is then a maximum-entropy problem, which we solve using Orthant-wise Limited-memory Quasi-Newton.<sup>4</sup>

To construct our initial transducer, we first learn weights for the marginal distribution on surface sounds by training the max-ent system with only the bias features active. Next, we manually set weights (Table 1) for insertions and deletions, which do not appear on the surface, and for faithfulness features. Other features get an initial weight of 0.

## 5 Experiments

### 5.1 Dataset

Our corpus is a processed version of the Bernstein-Ratner corpus (Bernstein-Ratner, 1987) from CHILDES (MacWhinney, 2000), which contains orthographic transcriptions of parent-child dyads with infants aged 13-23 months. Brent and Cartwright (1996) created a phonemic version of this corpus by extracting all infant-directed utterances and converted them to a phonemic transcription using a dictionary. This version, which contains 9790 utterances (33399 tokens, 1321 types), is now standard for word segmentation, but contains no phonetic variability.

Since producing a close phonetic transcription of this data would be impractical, we instead construct an approximate phonetic version using information from the Buckeye corpus (Pitt et al., 2007). Buckeye is a corpus of adult-directed conversational American English, and has been phonetically transcribed

<sup>4</sup>We use the implementation of Andrew and Gao (2007) with an  $l_2$  regularizer and regularization parameter  $r = 1$ ; although this could be tuned, in practice it has little effect on results.

Feature	Weight
output-is- $x$	marginal $p(x)$
output-is- $\epsilon$	0
same-sound	5
same- $\{\text{place, voice, manner}\}$	2
insertion	-3

Table 1: Initial transducer weights.

“about”	ahbawt:15, bawt:9, ihbawt:4, ahbawd:4, ihbawd:4, ahbaat:2, baw:1, ahbaht:1, erbawd:1, bawd:1, ahbaad:1, ahpaat:1, bah:1, baht:1, ah:1, ahbahd:1, ehbaat:1, ahbaed:1, ihbaht:1, baot:1
“wanna”	waanah:94, waanih:37, wahnah:16, waan:13, wahnah:8, wahnih:5, wahnay:3, waanlih:3, wehnih:2, waaneh:2, waonih:2, waaah:1, wuhnih:1, wahn:1, waantah:1, waanaa:1, wowih:1, waiih:1, wah:1, waaniy:1

Table 2: Empirical distribution of pronunciations of “about” and “wanna” in our dataset.

by hand to indicate realistic pronunciation variability. To create our phonetic corpus, we replace each phonemic word in the Bernstein-Ratner-Brent corpus with a phonetic pronunciation of that word sampled from the empirical distribution of pronunciations in Buckeye (Table 2). If the word never occurs in Buckeye, we use the original phonemic version.

Our corpus is not completely realistic as a sample of child-directed speech. Since each pronunciation is sampled independently, it lacks coarticulation and prosodic effects, and the distribution of pronunciations is derived from adult-directed rather than child-directed speech. Nonetheless, it represents phonetic variability more realistically than the Bernstein-Ratner-Brent corpus, while still maintaining the lexical characteristics of infant-directed speech (as compared to the Buckeye corpus, with its much larger vocabulary and more complex language model).

We conduct our development experiments on the first 8000 input utterances, holding out the remaining 1790 for evaluation. For evaluation experiments, we run the system on all 9790 utterances, reporting scores on only the last 1790.

## 5.2 Metrics

We evaluate our results by generalizing the three segmentation metrics from Goldwater et al. (2009): word boundary F-score, word token F-score, and lexicon (word type) F-score.

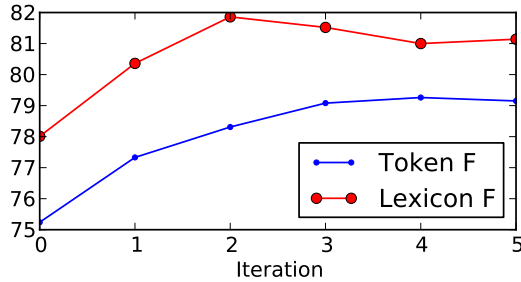


Figure 5: System scores over 5 iterations.

In our first set of experiments we evaluate how well our system clusters together surface forms derived from the same intended form, assuming gold standard word boundaries. We do not evaluate the induced intended forms directly against the gold standard intended forms—we want to evaluate cluster memberships and not labels. Instead we compute a one-to-one mapping between our induced lexical items and the gold standard, maximizing the agreement between the two (Haghighi and Klein, 2006). Using this mapping, we compute *mapped token F-score*<sup>5</sup> and *lexicon F-score*.

In our second set of experiments, we use unknown word boundaries and evaluate the segmentations. We report the standard word boundary F and *unlabeled word token F* as well as mapped F. The unlabeled token score counts correctly segmented tokens, whether assigned a correct intended form or not.

## 5.3 Known word boundaries

We first run our system with known word boundaries (Table 3). As a baseline, we treat every surface token as its own intended form (*none*). This baseline has fairly high accuracy; 65% of word tokens receive the most common pronunciation for their intended form.<sup>6</sup> As an upper bound, we find the best intended form for each surface type (*type unbound*). This correctly resolves 91% of tokens; the remaining error is due to homophones (surface types corresponding to more than one intended form). We also test our sys-

<sup>5</sup>When using the gold word boundaries, the precision and recall are equal and this is the same as the accuracy; in segmentation experiments the two differ, because with fewer segmentation boundaries, the system proposes fewer tokens. Only correctly segmented tokens which are also mapped to the correct form count as matches.

<sup>6</sup>The lexicon recall is not quite 100% because one rare word appears only as a homophone of another word.

System	Tok F	Lex P	Lex R	Lex F
none	65.4	50.2	99.7	66.7
initializer	75.2	83.2	73.3	78.0
system	79.2	87.1	75.9	81.1
oracle trans.	82.7	88.7	83.8	86.2
type ubound	91.0	97.5	98.0	97.7

Table 3: Results on 1790 utterances (known boundaries).

	Boundaries			Unlabeled Tokens		
	P	R	F	P	R	F
no var.	90.1	80.3	84.9	74.5	68.7	71.5
w/var.	70.4	93.5	80.3	56.5	69.7	62.4

Table 4: Degradation in `dpseg` segmentation performance caused by pronunciation variation.

	Mapped Tokens			Lexicon (types)		
	P	R	F	P	R	F
none	39.8	49.0	43.9	37.7	49.1	42.6
init	42.2	52.0	56.5	50.1	40.8	45.0
sys	44.2	54.5	48.8	48.6	43.1	45.7

Table 5: Results on 1790 utterances (induced boundaries).

tem using an oracle transducer (*oracle trans.*)—the transducer estimated from the upper-bound mapping. This scores 83%, showing that our articulatory feature set captures most, but not all, of the available information. At the beginning of bootstrapping, our system (*init*) scores 75%, but this improves to 79% after five iterations of reestimation (*system*). Most learning occurs in the first two or three iterations (Figure 5).

To determine the importance of different parts of our system, we run a few ablation tests on development data. Context information is critical to obtain a good solution; setting  $\Delta_L$  and  $\Delta_R$  to 0 lowers our dev token F-score from 83% to 75%. Initializing all feature weights to 0 yields a poor initial solution (18% dev token F instead of 75%), but after learning the result is only slightly lower than using the weights in Table 1 (78% rather than 80%), showing that the system is quite robust to initialization.

#### 5.4 Unknown word boundaries

As a simple extension of our model to the case of unknown word boundaries, we interleave it with an existing model of word segmentation, `dpseg` (Gold-

water et al., 2009).<sup>7</sup> In each iteration, we run the segmenter, then bootstrap our model for five iterations on the segmented output. We then concatenate the intended word sequence proposed by our model to produce the next iteration’s segmenter input.

Phonetic variation is known to reduce the performance of `dpseg` (Fleck, 2008; Boruta et al., 2011) and our experiments confirm this (Table 4). Using induced word boundaries also makes it harder to recover the lexicon (Table 5), lowering the baseline F-score from 67% to 43%. Nevertheless, our system improves the lexicon F-score to 46%, with token F rising from 44% to 49%, demonstrating the system’s ability to work without gold word boundaries. Unfortunately, performing multiple iterations between the segmenter and lexical-phonetic learner has little further effect; we hope to address this issue in future.

## 6 Conclusion

We have presented a noisy-channel model that simultaneously learns a lexicon, a bigram language model, and a model of phonetic variation, while using only the noisy surface forms as training data. It is the first model of lexical-phonetic acquisition to include word-level context and to be tested on an infant-directed corpus with realistic phonetic variability. Whether trained using gold standard or automatically induced word boundaries, the model recovers lexical items more effectively than a system that assumes no phonetic variability; moreover, the use of word-level context is key to the model’s success. Ultimately, we hope to extend the model to jointly infer word boundaries along with lexical-phonetic knowledge, and to work directly from acoustic input. However, we have already shown that lexical-phonetic learning from a broad-coverage corpus is possible, supporting the claim that infants acquire lexical and phonetic knowledge simultaneously.

## Acknowledgements

This work was supported by EPSRC grant EP/H050442/1 to the second author.

<sup>7</sup>`dpseg1.2` from <http://homepages.inf.ed.ac.uk/sgwater/resources.html>

## References

- Guillaume Aimetti. 2009. Modelling early language acquisition skills: Towards a general statistical learning mechanism. In *Proceedings of the Student Research Workshop at EACL*.
- Armen Allahverdyan and Aram Galstyan. 2011. Comparative analysis of Viterbi training and ML estimation for HMMs. In *Advances in Neural Information Processing Systems (NIPS)*.
- Cyril Allauzen, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri. 2007. OpenFst: A general and efficient weighted finite-state transducer library. In *Proceedings of the Ninth International Conference on Implementation and Application of Automata, (CIAA 2007)*, volume 4783 of *Lecture Notes in Computer Science*, pages 11–23. Springer. <http://www.openfst.org>.
- Galen Andrew and Jianfeng Gao. 2007. Scalable training of L1-regularized log-linear models. In *ICML '07*.
- Lalit Bahl, Raimo Bakis, Frederick Jelinek, and Robert Mercer. 1980. Language-model/acoustic-channel-model balance mechanism. Technical disclosure bulletin Vol. 23, No. 7b, IBM, December.
- Nan Bernstein-Ratner. 1987. The phonology of parent-child speech. In K. Nelson and A. van Kleeck, editors, *Children's Language*, volume 6. Erlbaum, Hillsdale, NJ.
- L. Boruta, S. Peperkamp, B. Crabbé, E. Dupoux, et al. 2011. Testing the robustness of online word segmentation: effects of linguistic diversity and phonetic variation. *ACL HLT 2011*, page 1.
- Michael Brent and Timothy Cartwright. 1996. Distributional regularity and phonotactic constraints are useful for segmentation. *Cognition*, 61:93–125.
- Michael R. Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105, February.
- R. Daland and J.B. Pierrehumbert. 2010. Learning diphone-based segmentation. *Cognitive Science*.
- Markus Dreyer, Jason R. Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 1080–1089, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Joris Driesen, Louis ten Bosch, and Hugo Van hamme. 2009. Adaptive non-negative matrix factorization in a computational model of language acquisition. In *Proceedings of Interspeech*.
- E. Dupoux, G. Beraud-Sudreau, and S. Sagayama. 2011. Templatic features for modeling phoneme acquisition. In *Proceedings of the 33rd Annual Cognitive Science Society*.
- Naomi Feldman, Thomas Griffiths, and James Morgan. 2009. Learning phonetic categories by learning a lexicon. In *Proceedings of the 31st Annual Conference of the Cognitive Science Society (CogSci)*.
- Naomi Feldman. 2011. *Interactions between word and speech sound categorization in language acquisition*. Ph.D. thesis, Brown University.
- Margaret M. Fleck. 2008. Lexicalized phonotactic word segmentation. In *Proceedings of ACL-08: HLT*, pages 130–138, Columbus, Ohio, June. Association for Computational Linguistics.
- Sharon Goldwater and Mark Johnson. 2003. Learning OT constraint rankings using a maximum entropy model. In J. Spenader, A. Eriksson, and Osten Dahl, editors, *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, pages 111–120, Stockholm. Stockholm University.
- Sharon Goldwater, Tom Griffiths, and Mark Johnson. 2006. Interpolating between types and tokens by estimating power-law generators. In *Advances in Neural Information Processing Systems (NIPS) 18*.
- Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. In *46th Annual Meeting of the ACL*, pages 398–406.
- Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA, June. Association for Computational Linguistics.
- Bruce Hayes and Colin Wilson. 2008. A maximum entropy model of phonotactics and phonotactic learning. *Linguistic Inquiry*, 39(3):379–440.
- Bruce Hayes. 2011. *Introductory Phonology*. John Wiley and Sons.
- A. Jansen, K. Church, and H. Hermansky. 2010. Towards spoken term discovery at scale with zero resources. In *Proceedings of Interspeech*, pages 1676–1679.
- R. Kneser and H. Ney. 1995. Improved backing-off for M-gram language modeling. In *Proc. ICASSP '95*, pages 181–184, Detroit, MI, May.
- B. MacWhinney. 2000. *The CHILDES Project: Tools for Analyzing Talk. Vol 2: The Database*. Lawrence Erlbaum Associates, Mahwah, NJ, 3rd edition.
- Fergus R. McInnes and Sharon Goldwater. 2011. Unsupervised extraction of recurring words from infant-directed speech. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*.
- A. S. Park and J. R. Glass. 2008. Unsupervised pattern discovery in speech. *IEEE Transactions on Audio, Speech and Language Processing*, 16:186–197.

- Fernando Pereira, Michael Riley, and Richard Sproat. 1994. Weighted rational transductions and their application to human language processing. In *HLT*.
- Mark A. Pitt, Laura Dilley, Keith Johnson, Scott Kiesling, William Raymond, Elizabeth Hume, and Eric Fosler-Lussier. 2007. Buckeye corpus of conversational speech (2nd release).
- Okko Räsänen. 2011. A computational model of word segmentation from continuous speech using transitional probabilities of atomic acoustic events. *Cognition*, 120(2):28.
- Anton Rytting. 2007. *Preserving Subsegmental Variation in Modeling Word Segmentation (Or, the Raising of Baby Mondegreen)*. Ph.D. thesis, The Ohio State University.
- Valentin I. Spitzkovsky, Hiyan Alshawi, Daniel Jurafsky, and Christopher D. Manning. 2010. Viterbi training improves unsupervised dependency parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 9–17, Uppsala, Sweden, July. Association for Computational Linguistics.
- D. Swingley. 2005. Statistical clustering and the contents of the infant vocabulary. *Cognitive Psychology*, 50:86–132.
- Yee Whye Teh. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 985–992, Sydney, Australia, July. Association for Computational Linguistics.
- G.K. Vallabha, J.L. McClelland, F. Pons, J.F. Werker, and S. Amano. 2007. Unsupervised learning of vowel categories from infant-directed speech. *Proceedings of the National Academy of Sciences*, 104(33):13273–13278.
- B. Varadarajan, S. Khudanpur, and E. Dupoux. 2008. Unsupervised learning of acoustic sub-word units. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 165–168. Association for Computational Linguistics.
- A. Venkataraman. 2001. A statistical model for word discovery in transcribed speech. *Computational Linguistics*, 27(3):351–372.

# Discriminative Pronunciation Modeling: A Large-Margin, Feature-Rich Approach

Hao Tang, Joseph Keshet, and Karen Livescu

Toyota Technological Institute at Chicago

Chicago, IL USA

{haotang, jkeshet, klivescu}@ttic.edu

## Abstract

We address the problem of learning the mapping between words and their possible pronunciations in terms of sub-word units. Most previous approaches have involved generative modeling of the distribution of pronunciations, usually trained to maximize likelihood. We propose a discriminative, feature-rich approach using large-margin learning. This approach allows us to optimize an objective closely related to a discriminative task, to incorporate a large number of complex features, and still do inference efficiently. We test the approach on the task of lexical access; that is, the prediction of a word given a phonetic transcription. In experiments on a subset of the Switchboard conversational speech corpus, our models thus far improve classification error rates from a previously published result of 29.1% to about 15%. We find that large-margin approaches outperform conditional random field learning, and that the Passive-Aggressive algorithm for large-margin learning is faster to converge than the Pegasos algorithm.

## 1 Introduction

One of the problems faced by automatic speech recognition, especially of conversational speech, is that of modeling the mapping between words and their possible pronunciations in terms of sub-word units such as phones. While pronunciation dictionaries provide each word's *canonical* pronunciation(s) in terms of phoneme strings, running speech often includes pronunciations that differ greatly from

the dictionary. For example, some pronunciations of “probably” in the Switchboard conversational speech database are [p r aa b iy], [p r aa l iy], [p r ay], and [p ow ih] (Greenberg et al., 1996). While some words (e.g., common words) are more prone to such variation than others, the effect is extremely general: In the phonetically transcribed portion of Switchboard, fewer than half of the word tokens are pronounced canonically (Fosler-Lussier, 1999). In addition, pronunciation variants sometimes include sounds not present in the dictionary at all, such as nasalized vowels (“can’t” → [k ae\_n n t]) or fricatives introduced due to incomplete consonant closures (“legal” → [l iy g\_fr ix l]).<sup>1</sup> This variation makes pronunciation modeling one of the major challenges facing speech recognition (McAllaster et al., 1998; Jurafsky et al., 2001; Saraçlar and Khudanpur, 2004; Bourlard et al., 1999).<sup>2</sup>

Most efforts to address the problem have involved either learning alternative pronunciations and/or their probabilities (Holter and Svendsen, 1999) or using phonetic transformation (substitution, insertion, and deletion) rules, which can come from linguistic knowledge or be learned from data (Riley et al., 1999; Hazen et al., 2005; Hutchinson and Droppo, 2011). These have produced some improvements in recognition performance. However, they also tend to cause additional confusability due to the introduction of additional homonyms (Fosler-

<sup>1</sup>We use the ARPAbet phonetic alphabet with additional diacritics, such as [ \_n ] for nasalization and [ \_fr ] for frication.

<sup>2</sup>This problem is separate from the grapheme-to-phoneme problem, in which pronunciations are predicted from a word's spelling; here, we assume the availability of a dictionary of canonical pronunciations as is usual in speech recognition.

Lussier et al., 2002). Some other alternatives are articulatory pronunciation models, in which words are represented as multiple parallel sequences of articulatory features rather than single sequences of phones, and which outperform phone-based models on some tasks (Livescu and Glass, 2004; Jyothi et al., 2011); and models for learning edit distances between dictionary and actual pronunciations (Ristad and Yianilos, 1998; Filali and Bilmes, 2005).

All of these approaches are generative—i.e., they provide distributions over possible pronunciations given the canonical one(s)—and they are typically trained by maximizing the likelihood over training data. In some recent work, discriminative approaches have been proposed, in which an objective more closely related to the task at hand is optimized. For example, (Vinyals et al., 2009; Korkmazskiy and Juang, 1997) optimize a minimum classification error (MCE) criterion to learn the weights (equivalently, probabilities) of alternative pronunciations for each word; (Schramm and Beyerlein, 2001) use a similar approach with discriminative model combination. In this work, the weighted alternatives are then used in a standard (generative) speech recognizer. In other words, these approaches optimize generative models using discriminative criteria.

We propose a general, flexible discriminative approach to pronunciation modeling, rather than discriminatively optimizing a generative model. We formulate a linear model with a large number of word-level and subword-level feature functions, whose weights are learned by optimizing a discriminative criterion. The approach is related to the recently proposed segmental conditional random field (SCRF) approach to speech recognition (Zweig et al., 2011). The main differences are that we optimize large-margin objective functions, which lead to sparser, faster, and better-performing models than conditional random field optimization in our experiments; and we use a large set of different feature functions tailored to pronunciation modeling.

In order to focus attention on the pronunciation model alone, our experiments focus on a task that measures only the mapping between words and subword units. Pronunciation models have in the past been tested using a variety of measures. For generative models, phonetic error rate of generated pronunciations (Venkataramani and Byrne, 2001) and

phone- or frame-level perplexity (Riley et al., 1999; Jyothi et al., 2011) are appropriate measures. For our discriminative models, we consider the task of *lexical access*; that is, prediction of a single word given its pronunciation in terms of sub-word units (Fissore et al., 1989; Jyothi et al., 2011). This task is also sometimes referred to as “pronunciation recognition” (Ristad and Yianilos, 1998) or “pronunciation classification” (Filali and Bilmes, 2005.) As we show below, our approach outperforms both traditional phonetic rule-based models and the best previously published results on our data set obtained with generative articulatory approaches.

## 2 Problem setting

We define a *pronunciation of a word* as a representation of the way it is produced by a speaker in terms of some set of linguistically meaningful sub-word units. A pronunciation can be, for example, a sequence of phones or multiple sequences of *articulatory features* such as nasality, voicing, and tongue and lip positions. For purposes of this paper, we will assume that a pronunciation is a single sequence of units, but the approach applies to other representations. We distinguish between two types of pronunciations of a word: (i) *canonical* pronunciations, the ones typically found in the dictionary, and (ii) *surface* pronunciations, the ways a speaker may actually produce the word. In the task of *lexical access* we are given a surface pronunciation of a word, and our goal is to predict the word.

Formally, we define a pronunciation as a sequence of sub-word units  $\bar{p} = (p_1, p_2, \dots, p_K)$ , where  $p_k \in \mathcal{P}$  for all  $1 \leq k \leq K$  and  $\mathcal{P}$  is the set of all sub-word units. The index  $k$  can represent either a fixed-length frame or a variable-length segment.  $\mathcal{P}^*$  denotes the set of all finite-length sequences over  $\mathcal{P}$ . We denote a word by  $w \in \mathcal{V}$  where  $\mathcal{V}$  is the vocabulary. Our goal is to find a function  $f : \mathcal{P}^* \rightarrow \mathcal{V}$  that takes as input a surface pronunciation and returns the word from the vocabulary that was spoken.

In this paper we propose a discriminative supervised learning approach for learning the function  $f$  from a training set of pairs  $(\bar{p}, w)$ . We aim to find a function  $f$  that performs well on the training set as well as on unseen examples. Let  $\hat{w} = f(\bar{p})$  be the predicted word given the pronunciation  $\bar{p}$ . We assess the quality of the function  $f$  by the *zero-one loss*: if

$w \neq \hat{w}$  then the error is one, otherwise the error is zero. The goal of the learning process is to minimize the expected zero-one loss, where the expectation is taken with respect to a fixed but unknown distribution over words and surface pronunciations. In the next section we present a learning algorithm that aims to minimize the expected zero-one loss.

### 3 Algorithm

Similarly to previous work in structured prediction (Taskar et al., 2003; Tsochantaridis et al., 2005), we construct the function  $f$  from a predefined set of  $N$  feature functions,  $\{\phi_j\}_{j=1}^N$ , each of the form  $\phi_j : \mathcal{P}^* \times \mathcal{V} \rightarrow \mathbb{R}$ . Each feature function takes a surface pronunciation  $\bar{p}$  and a proposed word  $w$  and returns a scalar which, intuitively, should be correlated with whether the pronunciation  $\bar{p}$  corresponds to the word  $w$ . The feature functions map pronunciations of different lengths along with a proposed word to a vector of fixed dimension in  $\mathbb{R}^N$ . For example, one feature function might measure the Levenshtein distance between the pronunciation  $\bar{p}$  and the canonical pronunciation of the word  $w$ . This feature function counts the minimum number of edit operations (insertions, deletions, and substitutions) that are needed to convert the surface pronunciation to the canonical pronunciation; it is low if the surface pronunciation is close to the canonical one and high otherwise.

The function  $f$  maximizes a score relating the word  $w$  to the pronunciation  $\bar{p}$ . We restrict ourselves to scores that are linear in the feature functions, where each  $\phi_j$  is scaled by a weight  $\theta_j$ :

$$\sum_{j=1}^N \theta_j \phi_j(\bar{p}, w) = \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\bar{p}, w),$$

where we have used vector notation for the feature functions  $\boldsymbol{\phi} = (\phi_1, \dots, \phi_N)$  and for the weights  $\boldsymbol{\theta} = (\theta_1, \dots, \theta_N)$ . Linearity is not a very strong restriction, since the feature functions can be arbitrarily non-linear. The function  $f$  is defined as the word  $w$  that maximizes the score,

$$f(\bar{p}) = \operatorname{argmax}_{w \in \mathcal{V}} \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\bar{p}, w).$$

Our goal in learning  $\boldsymbol{\theta}$  is to minimize the expected zero-one loss:

$$\boldsymbol{\theta}^* = \operatorname{argmin}_{\boldsymbol{\theta}} \mathbb{E}_{(\bar{p}, w) \sim \rho} [\mathbb{1}_{w \neq f(\bar{p})}],$$

where  $\mathbb{1}_{\pi}$  is 1 if predicate  $\pi$  holds and 0 otherwise, and where  $\rho$  is an (unknown) distribution from which the examples in our training set are sampled i.i.d. Let  $S = \{(\bar{p}_1, w_1), \dots, (\bar{p}_m, w_m)\}$  be the training set. Instead of working directly with the zero-one loss, which is non-smooth and non-convex, we use the surrogate hinge loss, which upper-bounds the zero-one loss:

$$L(\boldsymbol{\theta}, \bar{p}_i, w_i) = \max_{w \in \mathcal{V}} \left[ \mathbb{1}_{w_i \neq w} - \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\bar{p}_i, w_i) + \boldsymbol{\theta} \cdot \boldsymbol{\phi}(\bar{p}_i, w) \right]. \quad (1)$$

Finding the weight vector  $\boldsymbol{\theta}$  that minimizes the  $\ell^2$ -regularized average of this loss function is the structured support vector machine (SVM) problem (Taskar et al., 2003; Tsochantaridis et al., 2005):

$$\boldsymbol{\theta}^* = \operatorname{argmin}_{\boldsymbol{\theta}} \frac{\lambda}{2} \|\boldsymbol{\theta}\|^2 + \frac{1}{m} \sum_{i=1}^m L(\boldsymbol{\theta}, \bar{p}_i, w_i), \quad (2)$$

where  $\lambda$  is a user-defined tuning parameter that balances between regularization and loss minimization.

In practice, we have found that solving the quadratic optimization problem given in Eq. (2) converges very slowly using standard methods such as stochastic gradient descent (Shalev-Shwartz et al., 2007). We use a slightly different algorithm, the Passive-Aggressive (PA) algorithm (Crammer et al., 2006), whose average loss is comparable to that of the structured SVM solution (Keshet et al., 2007).

The Passive-Aggressive algorithm is an efficient online algorithm that, under some conditions, can be viewed as a dual-coordinate ascent minimizer of Eq. (2) (The connection to dual-coordinate ascent can be found in (Hsieh et al., 2008)). The algorithm begins by setting  $\boldsymbol{\theta} = \mathbf{0}$  and proceeds in rounds. In the  $t$ -th round the algorithm picks an example  $(\bar{p}_i, w_i)$  from  $S$  at random uniformly without replacement. Denote by  $\boldsymbol{\theta}^{t-1}$  the value of the weight vector before the  $t$ -th round. Let  $\hat{w}_i^t$  denote the predicted word for the  $i$ -th example according to  $\boldsymbol{\theta}^{t-1}$ :

$$\hat{w}_i^t = \operatorname{argmax}_{w \in \mathcal{V}} \boldsymbol{\theta}^{t-1} \cdot \boldsymbol{\phi}(\bar{p}_i, w) + \mathbb{1}_{w_i \neq w}.$$

Let  $\Delta \boldsymbol{\phi}_i^t = \boldsymbol{\phi}(\bar{p}_i, w_i) - \boldsymbol{\phi}(\bar{p}_i, \hat{w}_i^t)$ . Then the algorithm updates the weight vector  $\boldsymbol{\theta}^t$  as follows:

$$\boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1} + \alpha_i^t \Delta \boldsymbol{\phi}_i^t \quad (3)$$



where

$$\alpha_i^t = \min \left\{ \frac{1}{\lambda m}, \frac{\mathbb{1}_{w_i \neq \hat{w}_i^t} - \boldsymbol{\theta} \cdot \Delta \phi_i^t}{\|\Delta \phi_i^t\|} \right\}.$$

In practice we iterate over the  $m$  examples in the training set several times; each such iteration is an *epoch*. The final weight vector is set to the average over all weight vectors during training.

An alternative loss function that is often used to solve structured prediction problems is the *log-loss*:

$$L(\boldsymbol{\theta}, \bar{p}_i, w_i) = -\log P_{\boldsymbol{\theta}}(w_i | \bar{p}_i) \quad (4)$$

where the probability is defined as

$$P_{\boldsymbol{\theta}}(w_i | \bar{p}_i) = \frac{e^{\boldsymbol{\theta} \cdot \phi(\bar{p}_i, w_i)}}{\sum_{w \in \mathcal{V}} e^{\boldsymbol{\theta} \cdot \phi(\bar{p}_i, w)}}.$$

Minimization of Eq. (2) under the log-loss results in a probabilistic model commonly known as a conditional random field (CRF) (Lafferty et al., 2001). By taking the sub-gradient of Eq. (4), we can obtain an update rule similar to the one shown in Eq. (3).

## 4 Feature functions

Before defining the feature functions, we define some notation. Suppose  $\bar{p} \in \mathcal{P}^*$  is a sequence of sub-word units. We use  $\bar{p}_{1:n}$  to denote the  $n$ -gram substring  $p_1 \dots p_n$ . The two substrings  $\bar{a}$  and  $\bar{b}$  are said to be equal if they have the same length and  $a_i = b_i$  for  $1 \leq i \leq n$ . For a given sub-word unit  $n$ -gram  $\bar{u} \in \mathcal{P}^n$ , we use the shorthand  $\bar{u} \in \bar{p}$  to mean that we can find  $\bar{u}$  in  $\bar{p}$ ; i.e., there exists an index  $i$  such that  $\bar{p}_{i:i+n} = \bar{u}$ . We use  $|\bar{p}|$  to denote the length of the sequence  $\bar{p}$ .

We assume we have a pronunciation dictionary, which is a set of words and their baseforms. We access the dictionary through the function *pron*, which takes a word  $w \in \mathcal{V}$  and returns a set of baseforms.

### 4.1 TF-IDF feature functions

Term frequency (TF) and inverse document frequency (IDF) are measures that have been heavily used in information retrieval to search for documents using word queries (Salton et al., 1975). Similarly to (Zweig et al., 2010), we adapt TF and IDF by treating a sequence of sub-word units as a “document” and  $n$ -gram sub-sequences as “words.” In this analogy, we use sub-sequences in surface pronunciations to “search” for baseforms in the dictionary. These

features measure the frequency of each  $n$ -gram in observed pronunciations of a given word in the training set, along with the discriminative power of the  $n$ -gram. These features are therefore only meaningful for words actually observed in training.

The term frequency of a sub-word unit  $n$ -gram  $\bar{u} \in \mathcal{P}^n$  in a sequence  $\bar{p}$  is the length-normalized frequency of the  $n$ -gram in the sequence:

$$\text{TF}_{\bar{u}}(\bar{p}) = \frac{1}{|\bar{p}| - |\bar{u}| + 1} \sum_{i=1}^{|\bar{p}| - |\bar{u}| + 1} \mathbb{1}_{\bar{u} = \bar{p}_{i:i+|\bar{u}|-1}}.$$

Next, define the set of words in the training set that contain the  $n$ -gram  $\bar{u}$  as  $\mathcal{V}_{\bar{u}} = \{w \in \mathcal{V} \mid (\bar{p}, w) \in S, \bar{u} \in \bar{p}\}$ . The inverse document frequency (IDF) of an  $n$ -gram  $\bar{u}$  is defined as

$$\text{IDF}_{\bar{u}} = \log \frac{|\mathcal{V}|}{|\mathcal{V}_{\bar{u}}|}.$$

IDF represents the discriminative power of an  $n$ -gram: An  $n$ -gram that occurs in few words is better at word discrimination than a very common  $n$ -gram.

Finally, we define word-specific features using TF and IDF. Suppose the vocabulary is indexed:  $\mathcal{V} = \{w_1, \dots, w_n\}$ . Define  $\bar{e}_w$  as a binary vector with elements

$$(\bar{e}_w)_i = \mathbb{1}_{w_i = w}.$$

We define the TF-IDF feature function of  $\bar{u}$  as

$$\phi_{\bar{u}}(\bar{p}, w) = (\text{TF}_{\bar{u}}(\bar{p}) \times \text{IDF}_{\bar{u}}) \otimes \bar{e}_w,$$

where  $\otimes : \mathbb{R}^{a \times b} \times \mathbb{R}^{c \times d} \rightarrow \mathbb{R}^{ac \times bd}$  is the tensor product. We therefore have as many TF-IDF feature functions as we have  $n$ -grams. In practice, we only consider  $n$ -grams of a certain order (e.g., bigrams).

The following toy example demonstrates how the TF-IDF features are computed. Suppose we have  $\mathcal{V} = \{\text{problem, probably}\}$ . The dictionary maps “problem” to /pcl p r aa bcl b l ax m/ and “probably” to /pcl p r aa bcl b l iy/, and our input is  $(\bar{p}, w) = ([p r aa b l iy], \text{problem})$ . Then for the bigram /l iy/, we have  $\text{TF}_{/l iy/}(\bar{p}) = 1/5$  (one out of five bigrams in  $\bar{p}$ ), and  $\text{IDF}_{/l iy/} = \log(2/1)$  (one word out of two in the dictionary). The indicator vector is  $\bar{e}_{\text{problem}} = [1 \ 0]^T$ , so the final feature is

$$\phi_{/l iy/}(\bar{p}, w) = \begin{bmatrix} \frac{1}{5} \log \frac{2}{1} \\ 0 \end{bmatrix}.$$

## 4.2 Length feature function

The length feature functions measure how the length of a word’s surface form tends to deviate from the baseform. These functions are parameterized by  $a$  and  $b$  and are defined as

$$\phi_{a \leq \Delta \ell < b}(\bar{p}, w) = \mathbb{1}_{a \leq \Delta \ell < b} \otimes \bar{e}_w,$$

where  $\Delta \ell = |\bar{p}| - |\bar{v}|$ , for some baseform  $\bar{v} \in \text{pron}(w)$ . The parameters  $a$  and  $b$  can be either positive or negative, so the model can learn whether the surface pronunciations of a word tend to be longer or shorter than the baseform. Like the TF-IDF features, this feature is only meaningful for words actually observed in training.

As an example, suppose we have  $\mathcal{V} = \{\text{problem, probably}\}$ , and the word “probably” has two baseforms, /pcl p r aa bcl b l iy/ (of length eight) and /pcl p r aa bcl b ax bcl b l iy/ (of length eleven). If we are given an input  $(\bar{p}, w) = ([\text{pcl p r aa bcl l ax m}], \text{probably})$ , whose length of the surface form is eight, then the length features for the ranges  $0 \leq \Delta \ell < 1$  and  $-3 \leq \Delta \ell < -2$  are

$$\begin{aligned} \phi_{0 \leq \Delta \ell < 1}(\bar{p}, w) &= [0 \quad 1]^\top, \\ \phi_{-3 \leq \Delta \ell < -2}(\bar{p}, w) &= [0 \quad 1]^\top, \end{aligned}$$

respectively. Other length features are all zero.

## 4.3 Phonetic alignment feature functions

Beyond the length, we also measure specific phonetic deviations from the dictionary. We define phonetic alignment features that count the (normalized) frequencies of phonetic insertions, phonetic deletions, and substitutions of one surface phone for another baseform phone. Given  $(\bar{p}, w)$ , we use dynamic programming to align the surface form  $\bar{p}$  with all of the baseforms of  $w$ . Following (Riley et al., 1999), we encode a phoneme/phone with a 4-tuple: consonant manner, consonant place, vowel manner, and vowel place. Let the dash symbol “-” be a gap in the alignment (corresponding to an insertion/deletion). Given  $p, q \in \mathcal{P} \cup \{-\}$ , we say that a pair  $(p, q)$  is a deletion if  $p \in \mathcal{P}$  and  $q = -$ , is an insertion if  $p = -$  and  $q \in \mathcal{P}$ , and is a substitution if both  $p, q \in \mathcal{P}$ . Given  $p, q \in \mathcal{P} \cup \{-\}$ , let  $(s_1, s_2, s_3, s_4)$  and  $(t_1, t_2, t_3, t_4)$  be the corresponding 4-tuple encoding of  $p$  and  $q$ , respectively. The

	p	r	aa	pcl	p	er	l	iy	
pcl	p	r	aa	bcl	b	-	l	iy	
	p	r	aa	pcl	p	er	-	-	l iy
pcl	p	r	aa	bcl	b	ax	bcl	b	l iy

Table 1: Possible alignments of [p r aa pcl p er l iy] with two baseforms of “probably” in the dictionary.

similarity between  $p$  and  $q$  is defined as

$$s(p, q) = \begin{cases} 1, & \text{if } p = - \text{ or } q = -; \\ \sum_{i=1}^4 \mathbb{1}_{s_i=t_i}, & \text{otherwise.} \end{cases}$$

Consider aligning  $\bar{p}$  with the  $K_w = |\text{pron}(w)|$  baseforms of  $w$ . Define the length of the alignment with the  $k$ -th baseform as  $L_k$ , for  $1 \leq k \leq K_w$ . The resulting alignment is a sequence of pairs  $(a_{k,1}, b_{k,1}), \dots, (a_{k,L_k}, b_{k,L_k})$ , where  $a_{k,i}, b_{k,i} \in \mathcal{P} \cup \{-\}$  for  $1 \leq i \leq L_k$ . Now we define the alignment features, given  $p, q \in \mathcal{P} \cup \{-\}$ , as

$$\phi_{p \rightarrow q}(\bar{p}, w) = \frac{1}{Z_p} \sum_{k=1}^{K_w} \sum_{i=1}^{L_k} \mathbb{1}_{a_{k,i}=p, b_{k,i}=q},$$

where the normalization term is

$$Z_p = \begin{cases} \sum_{k=1}^{K_w} \sum_{i=1}^{L_k} \mathbb{1}_{a_{k,i}=p}, & \text{if } p \in \mathcal{P}; \\ |\bar{p}| \cdot K_w & \text{if } p = -. \end{cases}$$

The normalization for insertions differs from the normalization for substitutions and deletions, so that the resulting values always lie between zero and one.

As an example, consider the input pair  $(\bar{p}, w) = ([\text{p r aa pcl p er l iy}], \text{probably})$  and suppose there are two baseforms of the word “probably” in the dictionary. Let one possible alignments be the one shown in Table 1. Since /p/ occurs four times in the alignments and two of them are aligned to [b], the feature for  $p \rightarrow b$  is then  $\phi_{p \rightarrow b}(\bar{p}, w) = 2/4$ .

Unlike the TF-IDF feature functions and the length feature functions, the alignment feature functions can assign a non-zero score to words that are not seen at training time (but are in the dictionary), as long as there is a good alignment with their baseforms. The weights given to the alignment features are the analogue of substitution, insertion, and deletion rule probabilities in traditional phone-based pronunciation models such as (Riley et al., 1999); they can also be seen as a generalized version of the Levenshtein features of (Zweig et al., 2011).

#### 4.4 Dictionary feature function

The dictionary feature is an indicator of whether a pronunciation is an exact match to a baseform, which also generalizes to words unseen in training. We define the dictionary feature as

$$\phi_{\text{dict}}(\bar{p}, w) = \mathbb{1}_{\bar{p} \in \text{pron}(w)}.$$

For example, assume there is a baseform /pɫ p r aa bɫ b l iy/ for the word “probably” in the dictionary, and  $\bar{p} = \text{/pɫ p r aa bɫ b l iy/}$ . Then  $\phi_{\text{dict}}(\bar{p}, \text{probably}) = 1$ , while  $\phi_{\text{dict}}(\bar{p}, \text{problem}) = 0$ .

#### 4.5 Articulatory feature functions

Articulatory models represented as dynamic Bayesian networks (DBNs) have been successful in the past on the lexical access task (Livescu and Glass, 2004; Jyothi et al., 2011). In such models, pronunciation variation is seen as the result of asynchrony between the articulators (lips, tongue, etc.) and deviations from the intended articulatory positions. Given a sequence  $\bar{p}$  and a word  $w$ , we use the DBN to produce an alignment at the articulatory level, which is a sequence of 7-tuples, representing the articulatory variables<sup>3</sup> lip opening, tongue tip location and opening, tongue body location and opening, velum opening, and glottis opening. We extract three kinds of features from the output—substitutions, asynchrony, and log-likelihood.

The substitution features are similar to the phonetic alignment features in Section 4.3, except that the alignment is not a sequence of pairs but a sequence of 14-tuples (7 for the baseform and 7 for the surface form). The DBN model is based on articulatory phonology (Browman and Goldstein, 1992), in which there are no insertions and deletions, only substitutions (apparent insertions and deletions are accounted for by articulatory asynchrony). Formally, consider the seven sets of articulatory variable values  $F_1, \dots, F_7$ . For example,  $F_1$  could be all of the values of lip opening,  $F_1 = \{\text{closed, critical, narrow, wide}\}$ . Let  $\mathcal{F} = \{F_1, \dots, F_7\}$ . Consider an articulatory variable  $F \in \mathcal{F}$ . Suppose the alignment for  $F$  is  $(a_1, b_1), \dots, (a_L, b_L)$ , where  $L$

<sup>3</sup>We use the term “articulatory variable” for the “articulatory features” of (Livescu and Glass, 2004; Jyothi et al., 2011), in order to avoid confusion with our feature functions.

is the length of the alignment and  $a_i, b_i \in F$ , for  $1 \leq i \leq L$ . Here the  $a_i$  are the intended articulatory variable values according to the baseform, and the  $b_i$  are the corresponding realized values. For each  $a, b \in F$  we define a substitution feature function:

$$\phi_{a \rightarrow b}(\bar{p}, w) = \frac{1}{L} \sum_{i=1}^L \mathbb{1}_{a_i=a, b_i=b}.$$

The asynchrony features are also extracted from the DBN alignments. Articulators are not always synchronized, which is one cause of pronunciation variation. We measure this by looking at the phones that two articulators are aiming to produce, and find the time difference between them. Formally, we consider two articulatory variables  $F_h, F_k \in \mathcal{F}$ . Let the alignment between the two variables be  $(a_1, b_1), \dots, (a_L, b_L)$ , where now  $a_i \in F_h$  and  $b_i \in F_k$ . Each  $a_i$  and  $b_i$  can be mapped back to the corresponding phone index  $t_{h,i}$  and  $t_{k,i}$ , for  $1 \leq i \leq L$ . The average degree of asynchrony is then defined as

$$\text{async}(F_h, F_k) = \frac{1}{L} \sum_{i=1}^L (t_{h,i} - t_{k,i}).$$

More generally, we compute the average asynchrony between any two sets of variables  $\mathcal{F}_1, \mathcal{F}_2 \subset \mathcal{F}$  as

$$\text{async}(\mathcal{F}_1, \mathcal{F}_2) = \frac{1}{L} \sum_{i=1}^L \left[ \frac{1}{|\mathcal{F}_1|} \sum_{F_h \in \mathcal{F}_1} t_{h,i} - \frac{1}{|\mathcal{F}_2|} \sum_{F_k \in \mathcal{F}_2} t_{k,i} \right].$$

We then define the asynchrony features as

$$\phi_{a \leq \text{async}(\mathcal{F}_1, \mathcal{F}_2) \leq b} = \mathbb{1}_{a \leq \text{async}(\mathcal{F}_1, \mathcal{F}_2) \leq b}.$$

Finally, the log-likelihood feature is the DBN alignment score, shifted and scaled so that the value lies between zero and one,

$$\phi_{\text{dbn-LL}}(\bar{p}, w) = \frac{\mathcal{L}(\bar{p}, w) - h}{c},$$

where  $\mathcal{L}$  is the log-likelihood function of the DBN,  $h$  is the shift, and  $c$  is the scale.

Note that none of the DBN features are word-specific, so that they generalize to words in the dictionary that are unseen in the training set.

## 5 Experiments

All experiments are conducted on a subset of the Switchboard conversational speech corpus that has

been labeled at a fine phonetic level (Greenberg et al., 1996); these phonetic transcriptions are the input to our lexical access models. The data subset, phone set  $\mathcal{P}$ , and dictionary are the same as ones previously used in (Livescu and Glass, 2004; Jyothi et al., 2011). The dictionary contains 3328 words, consisting of the 5000 most frequent words in Switchboard, excluding ones with fewer than four phones in their baseforms. The baseforms use a similar, slightly smaller phone set (lacking, e.g., nasalization). We measure performance by error rate (ER), the proportion of test examples predicted incorrectly.

The TF-IDF features used in the experiments are based on phone bigrams. For all of the articulatory DBN features, we use the DBN from (Livescu, 2005) (the one in (Jyothi et al., 2011) is more sophisticated and may be used in future work). For the asynchrony features, the articulatory pairs are  $(\mathcal{F}_1, \mathcal{F}_2) \in \{(\{\text{tongue tip}\}, \{\text{tongue body}\}), (\{\text{lip opening}\}, \{\text{tongue tip, tongue body}\}), (\{\text{lip opening, tongue tip, tongue body}\}, \{\text{glottis, velum}\})\}$ , as in (Livescu, 2005). The parameters  $(a, b)$  of the length and asynchrony features are drawn from  $(a, b) \in \{(-3, -2), (-2, -1), \dots, (2, 3)\}$ .

We compare the CRF<sup>4</sup>, Passive-Aggressive (PA), and Pegasus learning algorithms. The regularization parameter  $\lambda$  is tuned on the development set. We run all three algorithms for multiple epochs and pick the best epoch based on development set performance.

For the first set of experiments, we use the same division of the corpus as in (Livescu and Glass, 2004; Jyothi et al., 2011) into a 2492-word training set, a 165-word development set, and a 236-word test set. To give a sense of the difficulty of the task, we test two simple baselines. One is a lexicon lookup: If the surface form is found in the dictionary, predict the corresponding word; otherwise, guess randomly. For a second baseline, we calculate the Levenshtein (0-1 edit) distance between the input pronunciation and each dictionary baseform, and predict the word corresponding to the baseform closest to the input. The results are shown in the first two rows of Table 2. We can see that, by adding just the Levenshtein distance, the error rate drops signifi-

<sup>4</sup>We use the term ‘‘CRF’’ since the learning algorithm corresponds to CRF learning, although the task is multiclass classification rather than a sequence or structure prediction task.

Model	ER
lexicon lookup (from (Livescu, 2005))	59.3%
lexicon + Levenshtein distance	41.8%
(Jyothi et al., 2011)	29.1%
CRF/DP+	21.5%
PA/DP+	<b>15.2%</b>
Pegasus/DP+	<b>14.8%</b>
PA/ALL	<b>15.2%</b>

Table 2: Lexical access error rates (ER) on the same data split as in (Livescu and Glass, 2004; Jyothi et al., 2011). Models labeled X/Y use learning algorithm X and feature set Y. The feature set DP+ contains TF-IDF, DP alignment, dictionary, and length features. The set ALL contains DP+ and the articulatory DBN features. The best results are in bold; the differences among them are insignificant (according to McNemar’s test with  $p = .05$ ).

icantly. However, both baselines do quite poorly.

Table 2 shows the best previous result on this data set from the articulatory model of Jyothi et al., which greatly improves over our baselines as well as over a much more complex phone-based model (Jyothi et al., 2011). The remaining rows of Table 2 give results with our feature functions and various learning algorithms. The best result for PA/DP+ (the PA algorithm using all features besides the DBN features) on the development set is with  $\lambda = 100$  and 5 epochs. Tested on the test set, this model improves over (Jyothi et al., 2011) by 13.9% absolute (47.8% relative). The best result for Pegasus with the same features on the development set is with  $\lambda = 0.01$  and 10 epochs. On the test set, this model gives a 14.3% absolute improvement (49.1% relative). CRF learning with the same features performs about 6% worse than the corresponding PA and Pegasus models.

The single-threaded running time for PA/DP+ and Pegasus/DP+ is about 40 minutes per epoch, measured on a dual-core AMD 2.4GHz CPU with 8GB of memory; for CRF, it takes about 100 minutes for each epoch, which is almost entirely because the weight vector  $\theta$  is less sparse with CRF learning. In the PA and Pegasus algorithms, we only update  $\theta$  for the most confusable word, while in CRF learning, we sum over all words. In our case, the number of non-zero entries in  $\theta$  for PA and Pegasus is around 800,000; for CRF, it is over 4,000,000. Though PA and Pegasus take roughly the same amount of time per epoch, Pegasus tends to require more epochs to

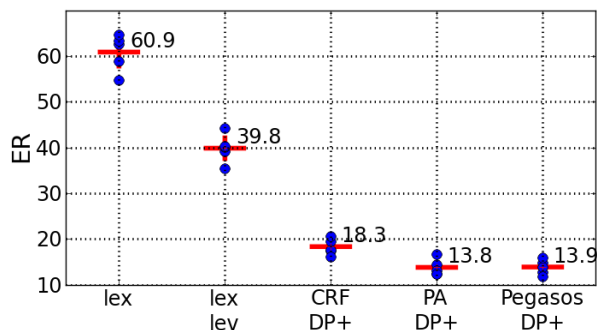


Figure 1: 5-fold cross validation (CV) results. The lexicon lookup baseline is labeled *lex*; *lex + lev* = lexicon lookup with Levenshtein distance. Each point corresponds to the test set error rate for one of the 5 data splits. The horizontal red line marks the mean of the results with means labeled, and the vertical red line indicates the mean plus and minus one standard deviation.

achieve the same performance as PA.

For the second experiment, we perform 5-fold cross-validation. We combine the training, development, and test sets from the previous experiment, and divide the data into five folds. We take three folds for training, one fold for tuning  $\lambda$  and the best epoch, and the remaining fold for testing. The results on the test fold are shown in Figure 1, which compares the learning algorithms, and Figure 2, which compares feature sets. Overall, the results are consistent with our first experiment. The feature selection experiments in Figure 2 shows that the TF-IDF features alone are quite weak, while the dynamic programming alignment features alone are quite good. Combining the two gives close to our best result. Although the marginal improvement gets smaller as we add more features, in general performance keeps improving the more features we add.

## 6 Discussion

The results in Section 5 are the best obtained thus far on the lexical access task on this conversational data set. Large-margin learning, using the Passive-Aggressive and Pegasus algorithms, has benefits over CRF learning for our task: It produces sparser models, is faster, and produces better lexical access results. In addition, the PA algorithm is faster than Pegasus on our task, as it requires fewer epochs.

Our ultimate goal is to incorporate such models into complete speech recognizers, that is to predict word sequences from acoustics. This requires (1)

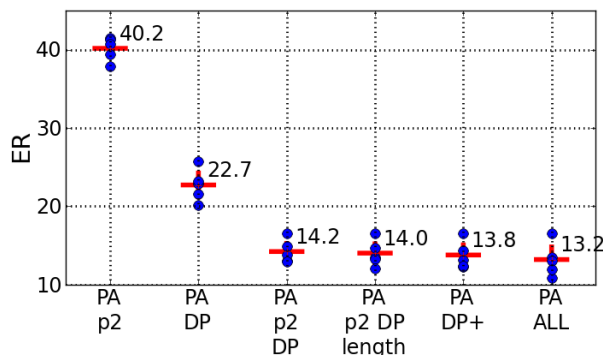


Figure 2: Feature selection results for five-fold cross validation. In the figure, phone bigram TF-IDF is labeled *p2*; phonetic alignment with dynamic programming is labeled *DP*. The dots and lines are as defined in Figure 1.

extension of the model and learning algorithm to word sequences and (2) feature functions that relate acoustic measurements to sub-word units. The extension to sequences can be done analogously to segmental conditional random fields (SCRFs). The main difference between SCRFs and our approach would be the large-margin learning, which can be straightforwardly applied to sequences. To incorporate acoustics, we can use feature functions based on classifiers of sub-word units, similarly to previous work on CRF-based speech recognition (Gunawardana et al., 2005; Morris and Fosler-Lussier, 2008; Prabhavalkar et al., 2011). Richer, longer-span (e.g., word-level) feature functions are also possible.

Thus far we have restricted the pronunciation-to-word score to linear combinations of feature functions. This can be extended to non-linear combinations using a kernel. This may be challenging in a high-dimensional feature space. One possibility is to approximate the kernels as in (Keshet et al., 2011). Additional extensions include new feature functions, such as context-sensitive alignment features, and joint inference and learning of the alignment models embedded in the feature functions.

## Acknowledgments

We thank Raman Arora, Arild Næss, and the anonymous reviewers for helpful suggestions. This research was supported in part by NSF grant IIS-0905633. The opinions expressed in this work are those of the authors and do not necessarily reflect the views of the funding agency.

## References

- H. Bourlard, S. Furui, N. Morgan, and H. Strik. 1999. Special issue on modeling pronunciation variation for automatic speech recognition. *Speech Communication*, 29(2-4).
- C. P. Browman and L. Goldstein. 1992. Articulatory phonology: an overview. *Phonetica*, 49(3-4).
- K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive aggressive algorithms. *Journal of Machine Learning Research*, 7.
- K. Filali and J. Bilmes. 2005. A dynamic Bayesian framework to model context and memory in edit distance learning: An application to pronunciation classification. In *Proc. Association for Computational Linguistics (ACL)*.
- L. Fissore, P. Laface, G. Micca, and R. Pieraccini. 1989. Lexical access to large vocabularies for speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(8).
- E. Fosler-Lussier, I. Amdal, and H.-K. J. Kuo. 2002. On the road to improved lexical confusability metrics. In *ISCA Tutorial and Research Workshop (ITRW) on Pronunciation Modeling and Lexicon Adaptation for Spoken Language Technology*.
- J. E. Fosler-Lussier. 1999. *Dynamic Pronunciation Models for Automatic Speech Recognition*. Ph.D. thesis, U. C. Berkeley.
- S. Greenberg, J. Hollenback, and D. Ellis. 1996. Insights into spoken language gleaned from phonetic transcription of the Switchboard corpus. In *Proc. International Conference on Spoken Language Processing (ICSLP)*.
- A. Gunawardana, M. Mahajan, A. Acero, and J. Platt. 2005. Hidden conditional random fields for phone classification. In *Proc. Interspeech*.
- T. J. Hazen, I. L. Hetherington, H. Shu, and K. Livescu. 2005. Pronunciation modeling using a finite-state transducer representation. *Speech Communication*, 46(2).
- T. Holter and T. Svendsen. 1999. Maximum likelihood modelling of pronunciation variation. *Speech Communication*.
- C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. 2008. A dual coordinate descent method for large-scale linear SVM. In *Proc. International Conference on Machine Learning (ICML)*.
- B. Hutchinson and J. Droppo. 2011. Learning non-parametric models of pronunciation. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- D. Jurafsky, W. Ward, Z. Jianping, K. Herold, Y. Xiyang, and Z. Sen. 2001. What kind of pronunciation variation is hard for triphones to model? In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- P. Jyothi, K. Livescu, and E. Fosler-Lussier. 2011. Lexical access experiments with context-dependent articulatory feature-based models. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan. 2007. A large margin algorithm for speech and audio segmentation. *IEEE Transactions on Acoustics, Speech, and Language Processing*, 15(8).
- J. Keshet, D. McAllester, and T. Hazan. 2011. PAC-Bayesian approach for minimization of phoneme error rate. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- F. Korkmazskiy and B.-H. Juang. 1997. Discriminative training of the pronunciation networks. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. International Conference on Machine Learning (ICML)*.
- K. Livescu and J. Glass. 2004. Feature-based pronunciation modeling with trainable asynchrony probabilities. In *Proc. International Conference on Spoken Language Processing (ICSLP)*.
- K. Livescu. 2005. *Feature-based Pronunciation Modeling for Automatic Speech Recognition*. Ph.D. thesis, Massachusetts Institute of Technology.
- D. McAllester, L. Gillick, F. Scatone, and M. Newman. 1998. Fabricating conversational speech data with acoustic models: A program to examine model-data mismatch. In *Proc. International Conference on Spoken Language Processing (ICSLP)*.
- J. Morris and E. Fosler-Lussier. 2008. Conditional random fields for integrating local discriminative classifiers. *IEEE Transactions on Acoustics, Speech, and Language Processing*, 16(3).
- R. Prabhavalkar, E. Fosler-Lussier, and K. Livescu. 2011. A factored conditional random field model for articulatory feature forced transcription. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.
- M. Riley, W. Byrne, M. Finke, S. Khudanpur, A. Ljolje, J. McDonough, H. Nock, M. Saraclar, C. Wooters, and G. Zavaliagos. 1999. Stochastic pronunciation modelling from hand-labelled phonetic corpora. *Speech Communication*, 29(2-4).
- E. S. Ristad and P. N. Yianilos. 1998. Learning string edit distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2).
- G. Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM*, 18.

- M. Saraçlar and S. Khudanpur. 2004. Pronunciation change in conversational speech and its implications for automatic speech recognition. *Computer Speech and Language*, 18(4).
- H. Schramm and P. Beyerlein. 2001. Towards discriminative lexicon optimization. In *Proc. Eurospeech*.
- S. Shalev-Shwartz, Y. Singer, and N. Srebro. 2007. Pegasos: Primal Estimated sub-GrAdient SOLver for SVM. In *Proc. International Conference on Machine Learning (ICML)*.
- B. Taskar, C. Guestrin, and D. Koller. 2003. Max-margin Markov networks. In *Advances in Neural Information Processing Systems (NIPS) 17*.
- I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6.
- V. Venkataramani and W. Byrne. 2001. MLLR adaptation techniques for pronunciation modeling. In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.
- O. Vinyals, L. Deng, D. Yu, and A. Acero. 2009. Discriminative pronunciation learning using phonetic decoder and minimum-classification-error criterion. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.
- G. Zweig, P. Nguyen, and A. Acero. 2010. Continuous speech recognition with a TF-IDF acoustic model. In *Proc. Interspeech*.
- G. Zweig, P. Nguyen, D. Van Compernelle, K. Demuynck, L. Atlas, P. Clark, G. Sell, M. Wang, F. Sha, H. Hermansky, D. Karakos, A. Jansen, S. Thomas, G.S.V.S. Sivaram, S. Bowman, and J. Kao. 2011. Speech recognition with segmental conditional random fields: A summary of the JHU CLSP 2010 summer workshop. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

# Discriminative Strategies to Integrate Multiword Expression Recognition and Parsing

**Matthieu Constant**

Université Paris-Est

LIGM, CNRS

France

mconstan@univ-mlv.fr

**Anthony Sigogne**

Université Paris-Est

LIGM, CNRS

France

sigogne@univ-mlv.fr

**Patrick Watrin**

Université de Louvain

CENTAL

Belgium

patrick.watrin@uclouvain.be

## Abstract

The integration of multiword expressions in a parsing procedure has been shown to improve accuracy in an artificial context where such expressions have been perfectly pre-identified. This paper evaluates two empirical strategies to integrate multiword units in a real constituency parsing context and shows that the results are not as promising as has sometimes been suggested. Firstly, we show that pre-grouping multiword expressions before parsing with a state-of-the-art recognizer improves multiword recognition accuracy and unlabeled attachment score. However, it has no statistically significant impact in terms of F-score as incorrect multiword expression recognition has important side effects on parsing. Secondly, integrating multiword expressions in the parser grammar followed by a reranker specific to such expressions slightly improves all evaluation metrics.

## 1 Introduction

The integration of Multiword Expressions (MWE) in real-life applications is crucial because such expressions have the particularity of having a certain level of idiomaticity. They form complex lexical units which, if they are considered, should significantly help parsing.

From a theoretical point of view, the integration of multiword expressions in the parsing procedure has been studied for different formalisms: Head-Driven Phrase Structure Grammar (Copestake *et al.*, 2002), Tree Adjoining Grammars (Schuler and Joshi, 2011), etc. From an empirical point of

view, their incorporation has also been considered such as in (Nivre and Nilsson, 2004) for dependency parsing and in (Arun and Keller, 2005) in constituency parsing. Although experiments always relied on a corpus where the MWEs were perfectly pre-identified, they showed that pre-grouping such expressions could significantly improve parsing accuracy. Recently, Green *et al.* (2011) proposed integrating the multiword expressions directly in the grammar without pre-recognizing them. The grammar was trained with a reference treebank where MWEs were annotated with a specific non-terminal node.

Our proposal is to evaluate two discriminative strategies in a real constituency parsing context: (a) pre-grouping MWE before parsing; this would be done with a state-of-the-art recognizer based on Conditional Random Fields; (b) parsing with a grammar including MWE identification and then reranking the output parses thanks to a Maximum Entropy model integrating MWE-dedicated features. (a) is the direct realistic implementation of the standard approach that was shown to reach the best results (Arun and Keller, 2005). We will evaluate if real MWE recognition (MWER) still positively impacts parsing, i.e., whether incorrect MWER does not negatively impact the overall parsing system. (b) is a more innovative approach to MWER (despite not being new in parsing): we select the final MWE segmentation after parsing in order to explore as many parses as possible (as opposed to method (a)). The experiments were carried out on the French Treebank (Abeillé *et al.*, 2003) where MWEs are annotated.



The paper is organized as follows: section 2 is an overview of the multiword expressions and their identification in texts; section 3 presents the two different strategies and their associated models; section 4 describes the resources used for our experiments (the corpus and the lexical resources); section 5 details the features that are incorporated in the models; section 6 reports on the results obtained.

## 2 Multiword expressions

### 2.1 Overview

Multiword expressions are lexical items made up of multiple lexemes that undergo idiosyncratic constraints and therefore offer a certain degree of idiomatity. They cover a wide range of linguistic phenomena: fixed and semi-fixed expressions, light verb constructions, phrasal verbs, named entities, etc. They may be contiguous (e.g. *traffic light*) or discontinuous (e.g. *John took your argument into account*). They are often divided into two main classes: multiword expressions defined through linguistic idiomatity criteria (*lexicalized phrases* in the terminology of Sag *et al.* (2002)) and those defined by statistical ones (i.e. simple collocations). Most linguistic criteria used to determine whether a combination of words is a MWE are based on syntactic and semantic tests such as the ones described in (Gross, 1986). For instance, the utterance *at night* is a MWE because it does display a strict lexical restriction (*\*at day*, *\*at afternoon*) and it does not accept any inserting material (*\*at cold night*, *\*at present night*). Such linguistically defined expressions may overlap with collocations which are the combinations of two or more words that cooccur more often than by chance. Collocations are usually identified through statistical association measures. A detailed description of MWEs can be found in (Baldwin and Nam, 2010).

In this paper, we focus on contiguous MWEs that form a lexical unit which can be marked by a part-of-speech tag (e.g. *at night* is an adverb, *because of* is a preposition). They can undergo limited morphological and lexical variations – e.g. *traffic (light+lights)*, *(apple+orange+...) juice* – and usually do not allow syntactic variations<sup>1</sup> such as inserts (e.g. *\*at*

<sup>1</sup>Such MWEs may very rarely accept inserts, often limited to single word modifiers: e.g. *in the short term*, *in the very short*

*cold night*). Such expressions can be analyzed at the lexical level. In what follows, we use the term *compounds* to denote such expressions.

### 2.2 Identification

The idiomatity property of MWEs makes them both crucial for Natural Language Processing applications and difficult to predict. Their actual identification in texts is therefore fundamental. There are different ways for achieving this objective. The simpler approach is lexicon-driven and consists in looking the MWEs up in an existing lexicon, such as in (Silberztein, 2000). The main drawback is that this procedure entirely relies on a lexicon and is unable to discover unknown MWEs. The use of collocation statistics is therefore useful. For instance, for each candidate in the text, Watrin and François (2011) compute on the fly its association score from an external ngram base learnt from a large raw corpus, and tag it as MWE if the association score is greater than a threshold. They reach excellent scores in the framework of a keyword extraction task. Within a validation framework (i.e. with the use of a reference corpus annotated in MWEs), Ramisch *et al.* (2010) developed a Support Vector Machine classifier integrating features corresponding to different collocation association measures. The results were rather low on the Genia corpus and Green *et al.* (2011) confirmed these bad results on the French Treebank. This can be explained by the fact that such a method does not make any distinctions between the different types of MWEs and the reference corpora are usually limited to certain types of MWEs. Furthermore, the lexicon-driven and collocation-driven approaches do not take the context into account, and therefore cannot discard some of the incorrect candidates. A recent trend is to couple MWE recognition with a linguistic analyzer: a POS tagger (Constant and Sigogne, 2011) or a parser (Green *et al.*, 2011). Constant and Sigogne (2011) trained a unified Conditional Random Fields model integrating different standard tagging features and features based on external lexical resources. They show a general tagging accuracy of 94% on the French Treebank. In terms of Multiword expression recognition, the accuracy was not

*term.*

clearly evaluated, but seemed to reach around 70-80% F-score. Green *et al.* (2011) proposed to include the MWER in the grammar of the parser. To do so, the MWEs in the training treebank were annotated with specific non-terminal nodes. They used a Tree Substitution Grammar instead of a Probabilistic Context-free Grammar (PCFG) with latent annotations in order to capture lexicalized rules as well as general rules. They showed that this formalism was more relevant to MWER than PCFG (71% F-score vs. 69.5%). Both methods have the advantage of being able to discover new MWEs on the basis of lexical and syntactic contexts. In this paper, we will take advantage of the methods described in this section by integrating them as features of a MWER model.

### 3 Two strategies, two discriminative models

#### 3.1 Pre-grouping Multiword Expressions

MWER can be seen as a sequence labelling task (like chunking) by using an IOB-like annotation scheme (Ramshaw and Marcus, 1995). This implies a theoretical limitation: recognized MWEs must be contiguous. The proposed annotation scheme is therefore theoretically weaker than the one proposed by Green *et al.* (2011) that integrates the MWER in the grammar and allows for discontinuous MWEs. Nevertheless, in practice, the compounds we are dealing with are very rarely discontinuous and if so, they solely contain a single word insert that can be easily integrated in the MWE sequence. Constant and Sigogne (2011) proposed to combine MWE segmentation and part-of-speech tagging into a single sequence labelling task by assigning to each token a tag of the form TAG+X where TAG is the part-of-speech (POS) of the lexical unit the token belongs to and X is either B (i.e. the token is at the beginning of the lexical unit) or I (i.e. for the remaining positions): *John/N+B hates/V+B traffic/N+B jams/N+I*. In this paper, as our task consists in jointly locating and tagging MWEs, we limited the POS tagging to MWEs only (TAG+B/TAG+I), simple words being tagged by O (outside): *John/O hates/O traffic/N+B jams/N+I*.

For such a task, we used Linear chain Conditional Random Fields (CRF) that are discriminative prob-

abilistic models introduced by Lafferty *et al.* (2001) for sequential labelling. Given an input sequence of tokens  $x = (x_1, x_2, \dots, x_N)$  and an output sequence of labels  $y = (y_1, y_2, \dots, y_N)$ , the model is defined as follows:

$$P_\lambda(y|x) = \frac{1}{Z(x)} \cdot \sum_t \sum_k \log \lambda_k \cdot f_k(t, y_t, y_{t-1}, x)$$

where  $Z(x)$  is a normalization factor depending on  $x$ . It is based on  $K$  features each of them being defined by a binary function  $f_k$  depending on the current position  $t$  in  $x$ , the current label  $y_t$ , the preceding one  $y_{t-1}$  and the whole input sequence  $x$ . The tokens  $x_i$  of  $x$  integrate the lexical value of this token but can also integrate basic properties which are computable from this value (for example: whether it begins with an upper case, it contains a number, its tags in an external lexicon, etc.). The feature is activated if a given configuration between  $t, y_t, y_{t-1}$  and  $x$  is satisfied (i.e.  $f_k(t, y_t, y_{t-1}, x) = 1$ ). Each feature  $f_k$  is associated with a weight  $\lambda_k$ . The weights are the parameters of the model, to be estimated. The features used for MWER will be described in section 5.

#### 3.2 Reranking

Discriminative reranking consists in reranking the  $n$ -best parses of a baseline parser with a discriminative model, hence integrating features associated with each node of the candidate parses. Charniak and Johnson (2005) introduced different features that showed significant improvement in general parsing accuracy (e.g. around +1 point in English). Formally, given a sentence  $s$ , the reranker selects the best candidate parse  $p$  among a set of candidates  $P(s)$  with respect to a scoring function  $V_\theta$ :

$$p^* = \operatorname{argmax}_{p \in P(s)} V_\theta(p)$$

The set of candidates  $P(s)$  corresponds to the  $n$ -best parses generated by the baseline parser. The scoring function  $V_\theta$  is the scalar product of a parameter vector  $\theta$  and a feature vector  $f$ :

$$V_\theta(p) = \theta \cdot f(p) = \sum_{j=1}^m \theta_j \cdot f_j(p)$$

where  $f_j(p)$  corresponds to the number of occurrences of the feature  $f_j$  in the parse  $p$ . According to

Charniak and Johnson (2005), the first feature  $f_1$  is the probability of  $p$  provided by the baseline parser. The vector  $\theta$  is estimated during the training stage from a reference treebank and the baseline parser outputs.

In this paper, we slightly deviate from the original reranker usage, by focusing on improving MWER in the context of parsing. Given the  $n$ -best parses, we want to select the one with the best MWE segmentation by keeping the overall parsing accuracy as high as possible. We therefore used MWE-dedicated features that we describe in section 5. The training stage was performed by using a Maximum entropy algorithm as in (Charniak and Johnson, 2005).

## 4 Resources

### 4.1 Corpus

The French Treebank<sup>2</sup> [FTB] (Abeillé *et al.*, 2003) is a syntactically annotated corpus made up of journalistic articles from *Le Monde* newspaper. We used the latest edition of the corpus (June 2010) that we preprocessed with the Stanford Parser preprocessing tools (Green *et al.*, 2011). It contains 473,904 tokens and 15,917 sentences. One benefit of this corpus is that its compounds are marked. Their annotation was driven by linguistic criteria such as the ones in (Gross, 1986). Compounds are identified with a specific non-terminal symbol "MWX" where X is the part-of-speech of the expression. They have a flat structure made of the part-of-speech of their components as shown in figure 1.

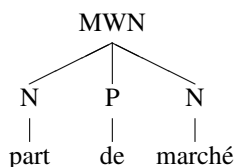


Figure 1: Subtree of MWE *part de marché* (market share): The MWN node indicates that it is a multiword noun; it has a flat internal structure N P N (noun – preposition – noun)

The French Treebank is composed of 435,860 lexical units (34,178 types). Among them, 5.3% are compounds (20.8% for types). In addition, 12.9%

<sup>2</sup><http://www.llf.cnrs.fr/Gens/Abeille/French-Treebank-fr.php>

of the tokens belong to a MWE, which, on average, has 2.7 tokens. The non-terminal tagset is composed of 14 part-of-speech labels and 24 phrasal ones (including 11 MWE labels). The train/dev/test split is the same as in (Green *et al.*, 2011): 1,235 sentences for test, 1,235 for development and 13,347 for training. The development and test sections are the same as those generally used for experiments in French, e.g. (Candito and Crabbé, 2009).

### 4.2 Lexical resources

French is a resource-rich language as attested by the existing morphological dictionaries which include compounds. In this paper, we use two large-coverage general-purpose dictionaries: Dela (Courtois, 1990; Courtois *et al.*, 1997) and Leff (Sagot, 2010). The Dela was manually developed in the 90's by a team of linguists. We used the distribution freely available in the platform Unitex<sup>3</sup> (Paumier, 2011). It is composed of 840,813 lexical entries including 104,350 multiword ones (91,030 multiword nouns). The compounds present in the resources respect the linguistic criteria defined in (Gross, 1986). The leff is a freely available dictionary<sup>4</sup> that has been automatically compiled by drawing from different sources and that has been manually validated. We used a version with 553,138 lexical entries including 26,311 multiword ones (22,673 multiword nouns). Their different modes of acquisition makes those two resources complementary. In both, lexical entries are composed of a inflected form, a lemma, a part-of-speech and morphological features. The Dela has an additional feature for most of the multiword entries: their syntactic surface form. For instance, *eau de vie* (brandy) has the feature NDN because it has the internal flat structure noun – preposition *de* – noun.

In order to compare compounds in these lexical resources with the ones in the French Treebank, we applied on the development corpus the dictionaries and the lexicon extracted from the training corpus. By a simple look-up, we obtained a preliminary lexicon-based MWE segmentation. The results are provided in table 1. They show that the use of external resources may improve recall, but they lead

<sup>3</sup><http://igm.univ-mlv.fr/~unitex>

<sup>4</sup><http://atoll.inria.fr/~sagot/leff.html>

to a decrease in precision as numerous MWEs in the dictionaries are not encoded as such in the reference corpus; in addition, the FTB suffers from some inconsistency in the MWE annotations.

	T	L	D	T+L	T+D	T+L+D
recall	75.9	31.7	59.0	77.3	83.4	84.0
precision	61.2	52.0	55.6	58.7	51.2	49.9
f-score	67.8	39.4	57.2	66.8	63.4	62.6

Table 1: Simple context-free application of the lexical resources on the development corpus: T is the MWE lexicon of the training corpus, L is the lefff, D is the Dela. The given scores solely evaluate MWE segmentation and not tagging.

In terms of statistical collocations, Watrin and François (2011) described a system that lists all the potential nominal collocations of a given sentence along with their association measure. The authors provided us with a list of 17,315 candidate nominal collocations occurring in the French treebank with their log-likelihood and their internal flat structure.

## 5 MWE-dedicated Features

The two discriminative models described in section 3 require MWE-dedicated features. In order to make these models comparable, we use two comparable sets of feature templates: one adapted to sequence labelling (CRF-based MWER) and the other one adapted to reranking (MaxEnt-based reranker). The MWER templates are instantiated at each position of the input sequence. The reranker templates are instantiated only for the nodes of the candidate parse tree, which are leaves dominated by a MWE node (i.e. the node has a MWE ancestor). We define a template  $T$  as follows:

- MWER: for each position  $n$  in the input sequence  $x$ ,

$$T = f(x, n)/y_n$$

- RERANKER: for each leaf (in position  $n$ ) dominated by a MWE node  $m$  in the current parse tree  $p$ ,

$$T = f(p, n)/label(m)/pos(p, n)$$

where  $f$  is a function to be defined;  $y_n$  is the output label at position  $n$ ;  $label(m)$  is the label of node  $m$  and  $pos(p, n)$  indicates the position of the word corresponding to  $n$  in the MWE sequence: B (starting position), I (remaining positions).

## 5.1 Endogenous Features

Endogenous features are features directly extracted from properties of the words themselves or from a tool learnt from the training corpus (e.g. a tagger).

**Word n-grams.** We use word unigrams and bigrams in order to capture multiwords present in the training section and to extract lexical cues to discover new MWEs. For instance, the bigram *coup de* is often the prefix of compounds such as *coup de pied* (kick), *coup de foudre* (love at first sight), *coup de main* (help).

**POS n-grams.** We use part-of-speech unigrams and bigrams in order to capture MWEs with irregular syntactic structures that might indicate the idiomacity of a word sequence. For instance, the POS sequence *preposition – adverb* associated with the compound *depuis peu* (recently) is very unusual in French. We also integrated mixed bigrams made up of a word and a part-of-speech.

**Specific features.** Due to their different use, each model integrates some specific features. In order to deal with unknown words and special tokens, we incorporate standard tagging features in the CRF: lowercase forms of the words, word prefixes of length 1 to 4, word suffixes of length 1 to 4, whether the word is capitalized, whether the token has a digit, whether it is an hyphen. We also add label bigrams. The reranker models integrate features associated with each MWE node, the value of which is the compound itself.

## 5.2 Exogenous Features

Exogenous features are features that are not entirely derived from the (reference) corpus itself. They are computed from external data (in our case, our lexical resources). The lexical resources might be useful to discover new expressions: usually, expressions that have standard syntax like nominal compounds and are difficult to predict from the endogenous features. The resources are applied to the corpus through a lexical analysis that generates, for each sentence, a finite-state automaton TFSA which represents all the possible analyses. The features are computed from the automaton TFSA.

**Lexicon-based features.** We associate each word with its part-of-speech tags found in our external morphological lexicon. All tags of a word constitute

an ambiguity class *ac*. If the word belongs to a compound, the compound tag is also incorporated in the ambiguity class. For instance, the word *night* (either a simple noun or a simple adjective) in the context *at night*, is associated with the class *adj\_noun\_adv+I* as it is located inside a compound adverb. This feature is directly computed from TFSA. The lexical analysis can lead to a preliminary MWE segmentation by using a shortest path algorithm that gives priority to compound analyses. This segmentation is also a source of features: a word belonging to a compound segment is assigned different properties such as the segment part-of-speech *mwt* and its syntactic structure *mws* encoded in the lexical resource, its relative position *mwpos* in the segment ('B' or 'I').

**Collocation-based features.** In our collocation resource, each candidate collocation of the French treebank is associated with its internal syntactic structure and its association score (log-likelihood). We divided these candidates into two classes: those whose score is greater than a threshold and the other ones. Therefore, a given word in the corpus can be associated with different properties whether it belongs to a potential collocation: the class *c* and the internal structure *cs* of the collocation it belongs to, its position *cpos* in the collocation (B: beginning; I: remaining positions; O: outside). We manually set the threshold to 150 after some tuning on the development corpus.

All feature templates are given in table 2.

Endogenous Features
$w(n+i), i \in \{-2, -1, 0, 1, 2\}$
$w(n+i)/w(n+i+1), i \in \{-2, -1, 0, 1\}$
$t(n+i), i \in \{-2, -1, 0, 1, 2\}$
$t(n+i)/t(n+i+1), i \in \{-2, -1, 0, 1\}$
$w(n+i)/t(n+j), (i, j) \in \{(1, 0), (0, 1), (-1, 0), (0, -1)\}$
Exogenous Features
$ac(n)$
$mwt(n)/mwpos(n)$
$mws(n)/mwpos(n)$
$c(n)/cs(n)/cpos(n)$

Table 2: Feature templates (*f*) used both in the MWER and the reranker models: *n* is the current position in the sentence,  $w(i)$  is the word at position *i*;  $t(i)$  is the part-of-speech tag of  $w(i)$ ; if the word at absolute position *i* is part of a compound in the Shortest Path Segmentation,  $mwt(i)$  and  $mws(i)$  are respectively the part-of-speech tag and the internal structure of the compound,  $mwpos(i)$  indicates its relative position in the compound (B or I).

## 6 Evaluation

### 6.1 Experiment Setup

We carried out 3 different experiments. We first tested a standalone MWE recognizer based on CRF. We then combined MWE pregrouping based on this recognizer and the Berkeley parser<sup>5</sup> (Petrov *et al.*, 2006) trained on the FTB where the compounds were concatenated (BKyc). Finally, we combined the Berkeley parser trained on the FTB where the compounds are annotated with specific non-terminals (BKY), and the reranker. In all experiments, we varied the set of features: *endo* are all endogenous features; *coll* and *lex* include all endogenous features plus collocation-based features and lexicon-based ones, respectively; *all* is composed of both endogenous and exogenous features. The CRF recognizer relies on the software *Wapiti*<sup>6</sup> (Lavergne *et al.*, 2010) to train and apply the model, and on the software *Unitex* (Paumier, 2011) to apply lexical resources. The part-of-speech tagger used to extract POS features was *lgtagger*<sup>7</sup> (Constant and Sigogne, 2011). To train the reranker, we used a MaxEnt algorithm<sup>8</sup> as in (Charniak and Johnson, 2005).

Results are reported using several standard measures, the  $F_1$  score, *unlabeled attachment* and *Leaf Ancestor* scores. The labeled  $F_1$  score [F1]<sup>9</sup>, defined by the standard protocol called PARSEVAL (Black *et al.*, 1991), takes into account the bracketing and labeling of nodes. The *unlabeled attachment score* [UAS] evaluates the quality of unlabeled

<sup>5</sup>We used the version adapted to French in the software *Bonsai* (Candito and Crabbé, 2009): [http://alpage.inria.fr/statgram/frdep/fr\\_stat\\_dep\\_parsing.html](http://alpage.inria.fr/statgram/frdep/fr_stat_dep_parsing.html). The original version is available at: <http://code.google.com/p/berkeleyparser/>. We trained the parser as follows: right binarization, no parent annotation, six split-merge cycles and default random seed initialisation (8).

<sup>6</sup>Wapiti can be found at <http://wapiti.limsi.fr/>. It was configured as follows: rprop algorithm, default L1-penalty value (0.5), default L2-penalty value (0.00001), default stopping criterion value (0.02%).

<sup>7</sup>Available at <http://igm.univ-mlv.fr/mconstan/research/software/>.

<sup>8</sup>We used the following mathematical libraries PETSc and TAO, freely available at <http://www.mcs.anl.gov/petsc/> and <http://www.mcs.anl.gov/research/projects/tao/>

<sup>9</sup>*Evalb* tool available at <http://nlp.cs.nyu.edu/evalb/>. We also used the evaluation by category implemented in the class *EvalbByCat* in the Stanford Parser.

dependencies between words of the sentence<sup>10</sup>. And finally, the *Leaf-Ancestor* score [LA]<sup>11</sup> (Sampson, 2003) computes the similarity between all paths (sequence of nodes) from each terminal node to the root node of the tree. The global score of a generated parse is equal to the average score of all terminal nodes. Punctuation tokens are ignored in all metrics. The quality of MWE identification was evaluated by computing the F<sub>1</sub> score on MWE nodes. We also evaluated the MWE segmentation by using the unlabeled F<sub>1</sub> score (U). In order to compare both approaches, parse trees generated by BKyc were automatically transformed in trees with the same MWE annotation scheme as the trees generated by BKY.

In order to establish the statistical significance of results between two parsing experiments in terms of F<sub>1</sub> and UAS, we used a unidirectional t-test for two independent samples<sup>12</sup>. The statistical significance between two MWE identification experiments was established by using the McNemar-s test (Gillick and Cox, 1989). The results of the two experiments are considered statistically significant with the computed value  $p < 0.01$ .

## 6.2 Standalone Multiword recognition

The results of the standalone MWE recognizer are given in table 3. They show that the lexicon-based system (*lex*) reaches the best score. Accuracy is improved by an absolute gain of +6.7 points as compared with BKY parser. The strictly endogenous system has a +4.9 point absolute gain, +5.4 points when collocations are added. That shows that most of the work is done by fully automatically acquired features (as opposed to features coming from a manually constructed lexicon). As expected, lexicon-based features lead to a 5.3 point recall improvement (with respect to non-lexicon based features) whereas precision is stable. The more precise system is the base one because it almost solely detects compounds present in the training corpus; nevertheless, it is unable to capture new MWEs (it has the

<sup>10</sup>This score is computed by using the tool available at <http://ilk.uvt.nl/conll/software.html>. The constituent trees are automatically converted into dependency trees with the tool *Bonsai*.

<sup>11</sup>*Leaf-ancestor assessment* tool available at <http://www.grsampson.net/Resources.html>

<sup>12</sup>Dan Bikel’s tool available at <http://www.cis.upenn.edu/~dbikel/software.html>.

lowest recall). BKY parser has the best recall among the non lexicon-based systems, i.e. it is the best one to discover new compounds as it is able to precisely detect irregular syntactic structures that are likely to be MWEs. Nevertheless, as it does not have a lexicalized strategy, it is not able to filter out incorrect candidates; the precision is therefore very low (the worst).

	P	R	F <sub>1</sub>	F <sub>1</sub> ≤ 40	U
base	<b>78.0</b>	68.3	72.8	71.2	74.3
endo	75.5	74.5	75.0	74.0	76.3
coll	76.6	74.4	75.5	74.9	77.0
lex	76.0	79.8	<b>77.8</b>	<b>77.8</b>	<b>79.0</b>
all	76.2	79.2	77.7	77.3	78.8
BKY	67.6	75.1	71.1	70.7	72.5
Stanford*	-	-	-	70.1	-
DP-TSG*	-	-	-	71.1	-

Table 3: MWE identification with CRF: *base* are the features corresponding to token properties and word n-grams. The differences between all systems are statistically significant with respect to McNemar’s test (Gillick and Cox, 1989), except *lex/all* and *all/coll*; *lex/coll* is "border-line". The results of the systems based on the Stanford Parser and the Tree Substitution Parser (DP-TSG) are reported from (Green *et al.*, 2011).

## 6.3 Combination of Multiword Expression Recognition and Parsing

We tested and compared the two proposed discriminative strategies by varying the sets of MWE-dedicated features. The results are reported in table 4. Table 5 compares the parsing systems, by showing the score differences between each of the tested system and the BKY parser.

Strat.	Feat.	Parser	F <sub>1</sub>	LA	UAS	F <sub>1</sub> (MWE)
-	-	BKY	80.61	92.91	82.99	71.1
pre	-	BKYc	75.47	91.10	76.74	0.0
pre	endo	BKYc	80.23	92.69	83.62	74.9
pre	coll	BKYc	80.32	92.73	83.77	75.5
pre	lex	BKYc	80.66	92.81	<b>84.16</b>	<b>77.4</b>
pre	all	BKYc	80.51	92.77	84.05	77.2
post	endo	BKY	80.87	92.94	83.49	72.9
post	coll	BKY	80.71	92.85	83.16	71.2
post	lex	BKY	<b>81.08</b>	<b>92.98</b>	83.98	74.5
post	all	BKY	81.03	92.96	83.97	74.3
pre	gold	BKYc	83.73	93.77	90.08	95.8

Table 4: Parsing evaluation: *pre* indicates a MWE pre-grouping strategy, whereas *post* is a reranking strategy with  $n = 50$ . The feature *gold* means that we have applied the parser on a gold MWE segmentation.

	$\Delta F_1$		$\Delta UAS$		$\Delta F_1(MWE)$	
	pre	post	pre	post	pre	post
endo	-0.38	+0.26	+0.63	+0.50	+3.8	+1.8
coll	-0.29	+0.10	+0.78	+0.17	+4.4	+0.1
lex	+0.05	+0.47	+1.17	+0.99	+6.3	+3.4

Table 5: Comparison of the strategies with respect to BKY parser.

Firstly, we note that the accuracy of the best realistic parsers is much lower than that of a parser with a golden MWE segmentation<sup>13</sup> (-2.65 and -5.92 respectively in terms of F-score and UAS), which shows the importance of not neglecting MWE recognition in the framework of parsing. Furthermore, pre-grouping has no statistically significant impact on the F-score<sup>14</sup>, whereas reranking leads to a statistically significant improvement (except for collocations). Both strategies also lead to a statistically significant UAS increase. Whereas both strategies improve the MWE recognition, pre-grouping is much more accurate (+2-4%); this might be due to the fact that an unlexicalized parser is limited in terms of compound identification, even within  $n$ -best analyses (cf. Oracle in table 6). The benefits of lexicon-based features are confirmed in this experiment, whereas the use of collocations in the reranking strategy seems to be rejected.

	endo	coll	lex	all	oracle
n=1	80.61 (71.1)				
n=5	80.74 (71.5)	<b>80.88</b> (71.7)	81.03 (73.4)	<b>81.05</b> (73.3)	83.17 (74.6)
n=20	<b>80.98</b> (72.9)	80.72 (70.6)	81.09 (73.6)	81.01 (73.0)	84.76 (75.5)
n=50	80.87 (72.9)	80.71 (71.2)	81.08 (74.5)	81.03 (74.3)	85.21 (76.4)
n=100	80.69 (72.0)	80.53 (70.0)	<b>81.12</b> (74.4)	80.93 (73.7)	<b>85.54</b> (76.4)

Table 6: Reranker  $F_1$  evaluation with respect to  $n$  and the types of features. The  $F_1(MWE)$  is given in parenthesis.

Table 7 shows the results by category. It indicates that both discriminative strategies are of interest in locating multiword adjectives, determiners and prepositions; the pre-grouping method appears to be particularly relevant for multiword nouns and

<sup>13</sup>The  $F_1(MWE)$  is not 100% with a golden segmentation because of tagging errors by the parser.

<sup>14</sup>Note that we observe an increase of +0.5 in  $F_1$  on the development corpus with lexicon-based features.

adverbs. However, it performs very poorly in multiword verb recognition. In terms of standard parsing accuracy, the pre-grouping approach has a very heterogeneous impact: Adverbial and Adjective Modifier phrases tend to be more accurate; verbal kernels and higher level constituents such as relative and subordinate clauses see their accuracy level drop, which shows that pre-recognition of MWE can have a negative impact on general parsing accuracy as MWE errors propagate to higher level constituents.

cat	#gold	BKY	endo (pre)	lex (pre)	endo (post)	lex (post)
MWET	4	0.0	N/A	N/A	N/A	N/A
MWA	22	37.2	+15.2	+21.3	+0.9	+4.7
MWV	47	62.1	-9.7	-13.2	+1.7	+2.5
MWD	24	62.1	+7.3	+10.2	0.0	+1.2
MWN	860	68.2	+4.0	+7.0	+1.7	+4.2
MWADV	357	72.1	+3.8	+6.4	+3.4	+4.1
MWPRO	31	84.2	-3.5	-0.9	0.0	0.0
MWP	294	79.1	+4.3	+5.8	+0.4	+1.1
MWC	86	85.7	+0.9	+3.7	+0.2	+1.0
Sint	209	47.2	-7.7	-8.7	+0.1	-0.2
AdP	86	48.8	+1.2	+3.0	+3.4	+5.1
Ssub	406	60.8	-1.1	-1.1	-0.3	-0.5
VPpart	541	63.2	-2.8	-2.1	-0.5	-1.6
Srel	408	74.8	-3.4	-3.5	-0.3	-0.6
VPinf	781	75.2	0.0	-0.1	-0.3	-0.3
COORD	904	75.2	+0.2	+0.4	-0.3	-0.4
PP	4906	76.7	-0.8	-0.3	+0.5	+0.7
AP	1482	74.5	+3.2	+3.9	+0.7	+1.6
NP	9023	79.8	-1.1	-0.8	+0.1	+0.2
VN	3089	94.0	-2.0	-1.0	0.0	0.0

Table 7: Evaluation by category with respect to BKY parser. The BKY column indicates the  $F_1$  of BKY parser.

## 7 Conclusions and Future Work

In this paper, we evaluated two discriminative strategies to integrate Multiword Expression Recognition in probabilistic parsing: (a) pre-grouping MWEs with a state-of-the-art recognizer and (b) MWE identification with a reranker after parsing. We showed that MWE pre-grouping significantly improves compound recognition and unlabeled dependency annotation, which implies that this strategy could be useful for dependency parsing. The reranking procedure evenly improves all evaluation scores. Future work could consist in combining both strategies: pre-grouping could suggest a set of potential MWE segmentations in order to make it more flexible for a parser; final decisions would then be made by the reranker.

## Acknowledgments

The authors are very grateful to Spence Green for his useful help on the treebank, and to Jennifer Thewissen for her careful proof-reading.

## References

- A. Abeillé and L. Clément and F. Toussenet. 2003. Building a treebank for French. *Treebanks*. In A. Abeillé (Ed.). Kluwer. Dordrecht.
- A. Arun and F. Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: The case of French. In *ACL*.
- E. Black, S. Abney, D. Flickinger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini and T. Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the DARPA Speech and Natural Language Workshop*.
- T. Baldwin and K.S. Nam. 2010. Multiword Expressions. *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group.
- M. -H. Candito and B. Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. *Proceedings of IWPT 2009*.
- E. Charniak and M. Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*.
- M. Constant and A. Sigogne. 2011. MWU-aware Part-of-Speech Tagging with a CRF model and lexical resources. In *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World (MWE'11)*.
- A. Copestake, F. Lambeau, A. Villavicencio, F. Bond, T. Baldwin, I. Sag, D. Flickinger. 2002. Multiword Expressions: Linguistic Precision and Reusability. *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*.
- B. Courtois. 1990. Un système de dictionnaires électroniques pour les mots simples du français. *Langue Française*. Vol. 87.
- B. Courtois, M. Garrigues, G. Gross, M. Gross, R. Jung, M. Mathieu-Colas, A. Monceaux, A. Poncet-Montange, M. Silberztein and R. Vivés. 1997. *Dictionnaire électronique DELAC : les mots composés binaires*. Technical Report. n. 56. LADL, University Paris 7.
- L. Gillick and S. Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. In *Proceedings of ICASSP'89*.
- S. Green, M.-C. de Marneffe, J. Bauer and C. D. Manning. 2011. Multiword Expression Identification with Tree Substitution Grammars: A Parsing tour de force with French. In *Empirical Method for Natural Language Processing (EMNLP'11)*.
- M. Gross. 1986. Lexicon Grammar. The Representation of Compound Words. In *Proceedings of Computational Linguistics (COLING'86)*.
- J. Lafferty and A. McCallum and F. Pereira. 2001. Conditional random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*.
- T. Lavergne, O. Cappé and F. Yvon. 2010. Practical Very Large Scale CRFs. In *ACL*.
- J. Nivre and J. Nilsson. 2004. Multiword units in syntactic parsing. In *Methodologies and Evaluation of Multiword Units in Real-World Applications (MEMURA)*.
- S. Paumier. 2011. Unitex 3.9 documentation. <http://igm.univ-mlv.fr/~unitex>.
- S. Petrov, L. Barrett, R. Thibaux and D. Klein. 2006. Learning accurate, compact and interpretable tree annotation. In *ACL*.
- C. Ramisch, A. Villavicencio and C. Boitet. 2010. mwe-toolkit: a framework for multiword expression identification. In *LREC*.
- L. A. Ramshaw and M. P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the 3rd Workshop on Very Large Corpora*.
- I. A. Sag, T. Baldwin, F. Bond, A. Copestake and D. Flickinger. 2002. Multiword Expressions: A Pain in the Neck for NLP. In *CICLING 2002*. Springer.
- B. Sagot. 2010. The Lefff, a freely available, accurate and large-coverage lexicon for French. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC'10)*.
- G. Sampson and A. Babarczy. 2003. A test of the leaf-ancestor metric for parsing accuracy. *Natural Language Engineering*. Vol. 9 (4).
- Seddah D., Candito M.-H. and Crabb B. 2009. Cross-parser evaluation and tagset variation: a French treebank study. *Proceedings of International Workshop on Parsing Technologies (IWPT'09)*.
- W. Schuler, A. Joshi. 2011. Tree-rewriting models of multi-word expressions. *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World (MWE'11)*.
- M. Silberztein. 2000. INTEX: an FST toolbox. *Theoretical Computer Science*, vol. 231(1).
- P. Watrin and T. François. 2011. N-gram frequency database reference to handle MWE extraction in NLP applications. In *Proceedings of the 2011 Workshop on MultiWord Expressions*.



# Utilizing Dependency Language Models for Graph-based Dependency Parsing Models

Wenliang Chen, Min Zhang\* and Haizhou Li

Human Language Technology, Institute for Infocomm Research, Singapore  
{wechen, mzhang, hli}@i2r.a-star.edu.sg

## Abstract

Most previous graph-based parsing models increase decoding complexity when they use high-order features due to exact-inference decoding. In this paper, we present an approach to enriching high-order feature representations for graph-based dependency parsing models using a dependency language model and beam search. The dependency language model is built on a large-amount of additional auto-parsed data that is processed by a baseline parser. Based on the dependency language model, we represent a set of features for the parsing model. Finally, the features are efficiently integrated into the parsing model during decoding using beam search. Our approach has two advantages. Firstly we utilize rich high-order features defined over a view of large scope and additional large raw corpus. Secondly our approach does not increase the decoding complexity. We evaluate the proposed approach on English and Chinese data. The experimental results show that our new parser achieves the best accuracy on the Chinese data and comparable accuracy with the best known systems on the English data.

## 1 Introduction

In recent years, there are many data-driven models that have been proposed for dependency parsing (McDonald and Nivre, 2007). Among them, graph-based dependency parsing models have achieved state-of-the-art performance for a wide range of languages as shown in recent CoNLL shared tasks

(Buchholz and Marsi, 2006; Nivre et al., 2007). In the graph-based models, dependency parsing is treated as a structured prediction problem in which the graphs are usually represented as factored structures. The information of the factored structures decides the features that the models can utilize. There are several previous studies that exploit high-order features that lead to significant improvements.

McDonald et al. (2005) and Covington (2001) develop models that represent first-order features over a single arc in graphs. By extending the first-order model, McDonald and Pereira (2006) and Carreras (2007) exploit second-order features over two adjacent arcs in second-order models. Koo and Collins (2010) further propose a third-order model that uses third-order features. These models utilize higher-order feature representations and achieve better performance than the first-order models. But this achievement is at the cost of the higher decoding complexity, from  $O(n^2)$  to  $O(n^4)$ , where  $n$  is the length of the input sentence. Thus, it is very hard to develop higher-order models further in this way.

How to enrich high-order feature representations without increasing the decoding complexity for graph-based models becomes a very challenging problem in the dependency parsing task. In this paper, we solve this issue by enriching the feature representations for a graph-based model using a dependency language model (DLM) (Shen et al., 2008). The  $N$ -gram DLM has the ability to predict the next child based on the  $N-1$  immediate previous children and their head (Shen et al., 2008). The basic idea behind is that we use the DLM to evaluate whether a valid dependency tree (McDonald and Nivre, 2007)

---

\*Corresponding author

is well-formed from a view of large scope. The parsing model searches for the final dependency trees by considering the original scores and the scores of DLM.

In our approach, the DLM is built on a large amount of auto-parsed data, which is processed by an original first-order parser (McDonald et al., 2005). We represent the features based on the DLM. The DLM-based features can capture the N-gram information of the parent-children structures for the parsing model. Then, they are integrated directly in the decoding algorithms using beam-search. Our new parsing model can utilize rich high-order feature representations but without increasing the complexity.

To demonstrate the effectiveness of the proposed approach, we conduct experiments on English and Chinese data. The results indicate that the approach greatly improves the accuracy. In summary, we make the following contributions:

- We utilize the dependency language model to enhance the graph-based parsing model. The DLM-based features are integrated directly into the beam-search decoder.
- The new parsing model uses the rich high-order features defined over a view of large scope and an additional large raw corpus, but without increasing the decoding complexity.
- Our parser achieves the best accuracy on the Chinese data and comparable accuracy with the best known systems on the English data.

## 2 Dependency language model

Language models play a very important role for statistical machine translation (SMT). The standard N-gram based language model predicts the next word based on the  $N - 1$  immediate previous words. However, the traditional N-gram language model can not capture long-distance word relations. To overcome this problem, Shen et al. (2008) proposed a dependency language model (DLM) to exploit long-distance word relations for SMT. The N-gram DLM predicts the next child of a head based on the  $N - 1$  immediate previous children and the head itself. In this paper, we define a DLM, which is similar to the one of Shen et al. (2008), to score entire dependency trees.

An input sentence is denoted by  $x = (x_0, x_1, \dots, x_i, \dots, x_n)$ , where  $x_0 = ROOT$  and does not depend on any other token in  $x$  and each token  $x_i$  refers to a word. Let  $y$  be a dependency tree for  $x$  and  $H(y)$  be a set that includes the words that have at least one dependent. For each  $x_h \in H(y)$ , we have a dependency structure  $D_h = (x_{Lk}, \dots, x_{L1}, x_h, x_{R1}, \dots, x_{Rm})$ , where  $x_{Lk}, \dots, x_{L1}$  are the children on the left side from the farthest to the nearest and  $x_{R1}, \dots, x_{Rm}$  are the children on the right side from the nearest to the farthest. Probability  $P(D_h)$  is defined as follows:

$$P(D_h) = P_L(D_h) \times P_R(D_h) \quad (1)$$

Here  $P_L$  and  $P_R$  are left and right side generative probabilities respectively. Suppose, we use a N-gram dependency language model.  $P_L$  is defined as follows:

$$\begin{aligned} P_L(D_h) \approx & P_{Lc}(x_{L1}|x_h) \\ & \times P_{Lc}(x_{L2}|x_{L1}, x_h) \\ & \times \dots \\ & \times P_{Lc}(x_{Lk}|x_{L(k-1)}, \dots, x_{L(k-N+1)}, x_h) \end{aligned} \quad (2)$$

where the approximation is based on the  $n$ th order Markov assumption. The right side probability is similar. For a dependency tree, we calculate the probability as follows:

$$P(y) = \prod_{x_h \in H(y)} P(D_h) \quad (3)$$

In this paper, we use a linear model to calculate the scores for the parsing models (defined in Section 3.1). Accordingly, we reform Equation 3. We define  $\mathbf{f}_{DLM}$  as a high-dimensional feature representation which is based on arbitrary features of  $P_{Lc}$ ,  $P_{Rc}$  and  $x$ . Then, the DLM score of tree  $y$  is in turn computed as the inner product of  $\mathbf{f}^{DLM}$  with a corresponding weight vector  $\mathbf{w}^{DLM}$ .

$$score^{DLM}(y) = \mathbf{f}^{DLM} \cdot \mathbf{w}^{DLM} \quad (4)$$

## 3 Parsing with dependency language model

In this section, we propose a parsing model which includes the dependency language model by extending the model of McDonald et al. (2005).

### 3.1 Graph-based parsing model

The graph-based parsing model aims to search for the maximum spanning tree (MST) in a graph (McDonald et al., 2005). We write  $(x_i, x_j) \in y$  if there is a dependency in tree  $y$  from word  $x_i$  to word  $x_j$  ( $x_i$  is the head and  $x_j$  is the dependent). A graph, denoted by  $G_x$ , consists of a set of nodes  $V_x = \{x_0, x_1, \dots, x_i, \dots, x_n\}$  and a set of arcs (edges)  $E_x = \{(x_i, x_j) | i \neq j, x_i \in V_x, x_j \in (V_x - x_0)\}$ , where the nodes in  $V_x$  are the words in  $x$ . Let  $T(G_x)$  be the set of all the subgraphs of  $G_x$  that are valid dependency trees (McDonald and Nivre, 2007) for sentence  $x$ .

The formulation defines the score of a dependency tree  $y \in T(G_x)$  to be the sum of the edge scores,

$$s(x, y) = \sum_{g \in y} \text{score}(\mathbf{w}, x, g) \quad (5)$$

where  $g$  is a spanning subgraph of  $y$ .  $g$  can be a single dependency or adjacent dependencies. Then  $y$  is represented as a set of factors. The model scores each factor using a weight vector  $\mathbf{w}$  that contains the weights for the features to be learned during training using the Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003; McDonald and Pereira, 2006). The scoring function is

$$\text{score}(\mathbf{w}, x, g) = \mathbf{f}(x, g) \cdot \mathbf{w} \quad (6)$$

where  $\mathbf{f}(x, g)$  is a high-dimensional feature representation which is based on arbitrary features of  $g$  and  $x$ .

The parsing model finds a *maximum spanning tree* (MST), which is the highest scoring tree in  $T(G_x)$ . The task of the decoding algorithm for a given sentence  $x$  is to find  $y^*$ ,

$$y^* = \arg \max_{y \in T(G_x)} s(x, y) = \arg \max_{y \in T(G_x)} \sum_{g \in y} \text{score}(\mathbf{w}, x, g)$$

### 3.2 Add DLM scores

In our approach, we consider the scores of the DLM when searching for the maximum spanning tree. Then for a given sentence  $x$ , we find  $y_{DLM}^*$ ,

$$y_{DLM}^* = \arg \max_{y \in T(G_x)} \left( \sum_{g \in y} \text{score}(\mathbf{w}, x, g) + \text{score}^{DLM}(y) \right)$$

After adding the DLM scores, the new parsing model can capture richer information. Figure 1 illustrates the changes. In the original first-order parsing model, we only utilize the information of single arc  $(x_h, x_{L(k-1)})$  for  $x_{L(k-1)}$  as shown in Figure 1-(a). If we use 3-gram DLM, we can utilize the additional information of the two previous children (nearer to  $x_h$  than  $x_{L(k-1)}$ ):  $x_{L(k-2)}$  and  $x_{L(k-3)}$  as shown in Figure 1-(b).

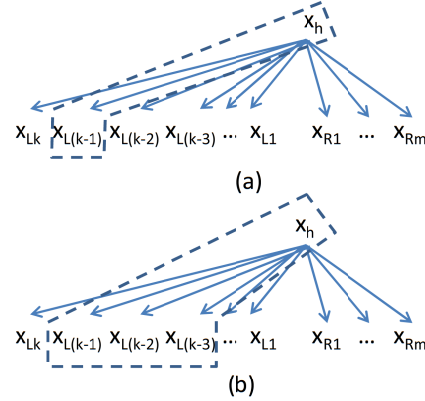


Figure 1: Adding the DLM scores to the parsing model

### 3.3 DLM-based feature templates

We define DLM-based features for  $D_h = (x_{Lk}, \dots, x_{L1}, x_h, x_{R1}, \dots, x_{Rm})$ . For each child  $x_{ch}$  on the left side, we have  $P_{Lc}(x_{ch} | \text{HIS})$ , where HIS refers to the  $N - 1$  immediate previous right children and head  $x_h$ . Similarly, we have  $P_{Rc}(x_{ch} | \text{HIS})$  for each child on the right side. Let  $P_u(x_{ch} | \text{HIS})$  ( $P_u(ch)$  in short) be one of the above probabilities. We use the map function  $\Phi(P_u(ch))$  to obtain the predefined discrete value (defined in Section 5.3). The feature templates are outlined in Table 1, where TYPE refers to one of the types:  $P_L$  or  $P_R$ , h\_pos refers to the part-of-speech tag of  $x_h$ , h\_word refers to the lexical form of  $x_h$ , ch\_pos refers to the part-of-speech tag of  $x_{ch}$ , and ch\_word refers to the lexical form of  $x_{ch}$ .

## 4 Decoding

In this section, we turn to the problem of adding the DLM in the decoding algorithm. We propose two ways: (1) Rescoring, in which we rescore the K-best list with the DLM-based features; (2) Intersect,

$\langle \Phi(P_u(ch)), \text{TYPE} \rangle$
$\langle \Phi(P_u(ch)), \text{TYPE}, h\_pos \rangle$
$\langle \Phi(P_u(ch)), \text{TYPE}, h\_word \rangle$
$\langle \Phi(P_u(ch)), \text{TYPE}, ch\_pos \rangle$
$\langle \Phi(P_u(ch)), \text{TYPE}, ch\_word \rangle$
$\langle \Phi(P_u(ch)), \text{TYPE}, h\_pos, ch\_pos \rangle$
$\langle \Phi(P_u(ch)), \text{TYPE}, h\_word, ch\_word \rangle$

Table 1: DLM-based feature templates

in which we add the DLM-based features in the decoding algorithm directly.

#### 4.1 Rescoring

We add the DLM-based features into the decoding procedure by using the rescoring technique used in (Shen et al., 2008). We can use an original parser to produce the K-best list. This method has the potential to be very fast. However, because the performance of this method is restricted to the K-best list, we may have to set K to a high number in order to find the best parsing tree (with DLM) or a tree acceptable close to the best (Shen et al., 2008).

#### 4.2 Intersect

Then, we add the DLM-based features in the decoding algorithm directly. The DLM-based features are generated online during decoding.

For our parser, we use the decoding algorithm of McDonald et al. (2005). The algorithm was extensions of the parsing algorithm of (Eisner, 1996), which was a modified version of the CKY chart parsing algorithm. Here, we describe how to add the DLM-based features in the first-order algorithm. The second-order and higher-order algorithms can be extended by the similar way.

The parsing algorithm independently parses the left and right dependents of a word and combines them later. There are two types of chart items (McDonald and Pereira, 2006): 1) a *complete item* in which the words are unable to accept more dependents in a certain direction; and 2) an *incomplete item* in which the words can accept more dependents in a certain direction. In the algorithm, we create both types of chart items with two directions for all the word pairs in a given sentence. The direction of a dependency is from the head to the dependent. The right (left) direction indicates the dependent is on the right (left) side of the head. Larger chart items are

created from pairs of smaller ones in a bottom-up style. In the following figures, complete items are represented by triangles and incomplete items are represented by trapezoids. Figure 2 illustrates the cubic parsing actions of the algorithm (Eisner, 1996) in the right direction, where  $s$ ,  $r$ , and  $t$  refer to the start and end indices of the chart items. In Figure 2-(a), all the items on the left side are complete and the algorithm creates the incomplete item (trapezoid on the right side) of  $s - t$ . This action builds a dependency relation from  $s$  to  $t$ . In Figure 2-(b), the item of  $s - r$  is incomplete and the item of  $r - t$  is complete. Then the algorithm creates the complete item of  $s - t$ . In this action, all the children of  $r$  are generated. In Figure 2, the longer vertical edge in a triangle or a trapezoid corresponds to the subroot of the structure (spanning chart). For example,  $s$  is the subroot of the span  $s - t$  in Figure 2-(a). For the left direction case, the actions are similar.

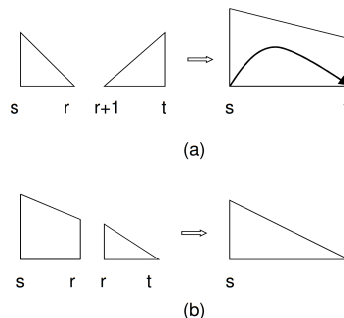


Figure 2: Cubic parsing actions of Eisner (Eisner, 1996)

Then, we add the DLM-based features into the parsing actions. Because the parsing algorithm is in the bottom-up style, the nearer children are generated earlier than the farther ones of the same head. Thus, we calculate the left or right side probability for a new child when a new dependency relation is built. For Figure 2-(a), we add the features of  $P_{Rc}(x_t|HIS)$ . Figure 3 shows the structure, where  $c_{Rs}$  refers to the current children (nearer than  $x_t$ ) of  $x_s$ . In the figure, HIS includes  $c_{Rs}$  and  $x_s$ .

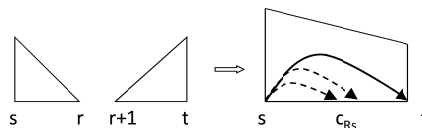


Figure 3: Add DLM-based features in cubic parsing

We use beam search to choose the one having the overall best score as the final parse, where  $K$  spans are built at each step (Zhang and Clark, 2008). At each step, we perform the parsing actions in the current beam and then choose the best  $K$  resulting spans for the next step. The time complexity of the new decoding algorithm is  $O(Kn^3)$  while the original one is  $O(n^3)$ , where  $n$  is the length of the input sentence. With the rich feature set in Table 1, the running time of Intersect is longer than the time of Rescoring. But Intersect considers more combination of spans with the DLM-based features than Rescoring that is only given a  $K$ -best list.

## 5 Implementation Details

### 5.1 Baseline parser

We implement our parsers based on the MSTParser<sup>1</sup>, a freely available implementation of the graph-based model proposed by (McDonald and Pereira, 2006). We train a first-order parser on the training data (described in Section 6.1) with the features defined in McDonald et al. (2005). We call this first-order parser Baseline parser.

### 5.2 Build dependency language models

We use a large amount of unannotated data to build the dependency language model. We first perform word segmentation (if needed) and part-of-speech tagging. After that, we obtain the word-segmented sentences with the part-of-speech tags. Then the sentences are parsed by the Baseline parser. Finally, we obtain the auto-parsed data.

Given the dependency trees, we estimate the probability distribution by relative frequency:

$$P_u(x_{ch}|\text{HIS}) = \frac{\text{count}(x_{ch}, \text{HIS})}{\sum_{x'_{ch}} \text{count}(x'_{ch}, \text{HIS})} \quad (7)$$

No smoothing is performed because we use the mapping function for the feature representations.

### 5.3 Mapping function

We can define different mapping functions for the feature representations. Here, we use a simple way. First, the probabilities are sorted in decreasing order. Let  $No(P_u(ch))$  be the position number of  $P_u(ch)$  in the sorted list. The mapping function is:

$$\Phi(P_u(ch)) = \begin{cases} PH & \text{if } No(P_u(ch)) \leq \text{TOP10} \\ PM & \text{if } \text{TOP10} < No(P_u(ch)) \leq \text{TOP30} \\ PL & \text{if } \text{TOP30} < No(P_u(ch)) \\ PO & \text{if } P_u(ch) = 0 \end{cases}$$

where TOP10 and TOP 30 refer to the position numbers of top 10% and top 30% respectively. The numbers, 10% and 30%, are tuned on the development sets in the experiments.

## 6 Experiments

We conducted experiments on English and Chinese data.

### 6.1 Data sets

For English, we used the Penn Treebank (Marcus et al., 1993) in our experiments. We created a standard data split: sections 2-21 for training, section 22 for development, and section 23 for testing. Tool ‘‘Penn2Malt’’<sup>2</sup> was used to convert the data into dependency structures using a standard set of head rules (Yamada and Matsumoto, 2003). Following the work of (Koo et al., 2008), we used the MXPOST (Ratnaparkhi, 1996) tagger trained on training data to provide part-of-speech tags for the development and the test set, and used 10-way jackknifing to generate part-of-speech tags for the training set. For the unannotated data, we used the BLLIP corpus (Charniak et al., 2000) that contains about 43 million words of WSJ text.<sup>3</sup> We used the MXPOST tagger trained on training data to assign part-of-speech tags and used the Baseline parser to process the sentences of the BLLIP corpus.

For Chinese, we used the Chinese Treebank (CTB) version 4.0<sup>4</sup> in the experiments. We also used the ‘‘Penn2Malt’’ tool to convert the data and created a data split: files 1-270 and files 400-931 for training, files 271-300 for testing, and files 301-325 for development. We used gold standard segmentation and part-of-speech tags in the CTB. The data partition and part-of-speech settings were chosen to match previous work (Chen et al., 2008; Yu et al., 2008; Chen et al., 2009). For the unannotated data, we used the XIN\_CMN portion of Chinese Gigaword<sup>5</sup> Version 2.0 (LDC2009T14) (Huang, 2009),

<sup>2</sup><http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html>

<sup>3</sup>We ensured that the text used for extracting subtrees did not include the sentences of the Penn Treebank.

<sup>4</sup><http://www.cis.upenn.edu/~chinese/>.

<sup>5</sup>We excluded the sentences of the CTB data from the Gigaword data

<sup>1</sup><http://mstparser.sourceforge.net>

which has approximately 311 million words whose segmentation and POS tags are given. We discarded the annotations due to the differences in annotation policy between CTB and this corpus. We used the MMA system (Kruengkrai et al., 2009) trained on the training data to perform word segmentation and POS tagging and used the Baseline parser to parse all the sentences in the data.

## 6.2 Features for basic and enhanced parsers

The previous studies have defined four types of features: (FT1) the first-order features defined in McDonald et al. (2005), (FT2SB) the second-order parent-siblings features defined in McDonald and Pereira (2006), (FT2GC) the second-order parent-child-grandchild features defined in Carreras (2007), and (FT3) the third-order features defined in (Koo and Collins, 2010).

We used the first- and second-order parsers of the MSTParser as the basic parsers. Then we enhanced them with other higher-order features using beam-search. Table 2 shows the feature settings of the systems, where MST1/2 refers to the basic first-/second-order parser and MSTB1/2 refers to the enhanced first-/second-order parser. MSTB1 and MSTB2 used the same feature setting, but used different order models. This resulted in the difference of using FT2SB (beam-search in MSTB1 vs exact-inference in MSTB2). We used these four parsers as the Baselines in the experiments.

System	Features
MST1	(FT1)
MSTB1	(FT1)+(FT2SB+FT2GC+FT3)
MST2	(FT1+FT2SB)
MSTB2	(FT1+FT2SB)+(FT2GC+FT3)

Table 2: Baseline parsers

We measured the parser quality by the unlabeled attachment score (UAS), i.e., the percentage of tokens (excluding all punctuation tokens) with the correct HEAD. In the following experiments, we used “Inter” to refer to the parser with Intersect, and “Rescore” to refer to the parser with Rescoring.

## 6.3 Development experiments

Since the setting of K (for beam search) affects our parsers, we studied its influence on the development

set for English. We added the DLM-based features to MST1. Figure 4 shows the UAS curves on the development set, where K is beam size for Intersect and K-best for Rescoring, the X-axis represents K, and the Y-axis represents the UAS scores. The parsing performance generally increased as the K increased. The parser with Intersect always outperformed the one with Rescoring.

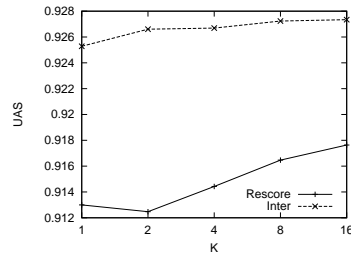


Figure 4: The influence of K on the development data

K	1	2	4	8	16
English	157.1	247.4	351.9	462.3	578.2

Table 3: The parsing times on the development set (seconds for all the sentences)

Table 3 shows the parsing times of Intersect on the development set for English. By comparing the curves of Figure 4, we can see that, while using larger K reduced the parsing speed, it improved the performance of our parsers. In the rest of the experiments, we set K=8 in order to obtain the high accuracy with reasonable speed and used Intersect to add the DLM-based features.

N	0	1	2	3	4
English	91.30	91.87	92.52	92.72	92.72
Chinese	87.36	87.96	89.33	89.92	90.40

Table 4: Effect of different N-gram DLMs

Then, we studied the effect of adding different N-gram DLMs to MST1. Table 4 shows the results. From the table, we found that the parsing performance roughly increased as the N increased. When N=3 and N=4, the parsers obtained the same scores for English. For Chinese, the parser obtained the best score when N=4. Note that the size of the Chinese unannotated data was larger than that of English. In the rest of the experiments, we used 3-gram for English and 4-gram for Chinese.

## 6.4 Main results on English data

We evaluated the systems on the testing data for English. The results are shown in Table 5, where -DLM refers to adding the DLM-based features to the Baselines. The parsers using the DLM-based features consistently outperformed the Baselines. For the basic models (MST1/2), we obtained absolute improvements of 0.94 and 0.63 points respectively. For the enhanced models (MSTB1/2), we found that there were 0.63 and 0.66 points improvements respectively. The improvements were significant in McNemar’s Test ( $p < 10^{-5}$ ) (Nivre et al., 2004).

Order1	UAS	Order2	UAS
MST1	90.95	MST2	91.71
MST-DLM1	91.89	MST-DLM2	92.34
MSTB1	91.92	MSTB2	92.10
MSTB-DLM1	92.55	MSTB-DLM2	92.76

Table 5: Main results for English

## 6.5 Main results on Chinese data

The results are shown in Table 6, where the abbreviations used are the same as those in Table 5. As in the English experiments, the parsers using the DLM-based features consistently outperformed the Baselines. For the basic models (MST1/2), we obtained absolute improvements of 4.28 and 3.51 points respectively. For the enhanced models (MSTB1/2), we got 3.00 and 2.93 points improvements respectively. We obtained large improvements on the Chinese data. The reasons may be that we use the very large amount of data and 4-gram DLM that captures high-order information. The improvements were significant in McNemar’s Test ( $p < 10^{-7}$ ).

Order1	UAS	Order2	UAS
MST1	86.38	MST2	88.11
MST-DLM1	90.66	MST-DLM2	91.62
MSTB1	88.38	MSTB2	88.66
MSTB-DLM1	91.38	MSTB-DLM2	91.59

Table 6: Main results for Chinese

## 6.6 Compare with previous work on English

Table 7 shows the performance of the graph-based systems that were compared, where McDonald06 refers to the second-order parser of McDonald

and Pereira (2006), Koo08-standard refers to the second-order parser with the features defined in Koo et al. (2008), Koo10-model1 refers to the third-order parser with model1 of Koo and Collins (2010), Koo08-dep2c refers to the second-order parser with cluster-based features of (Koo et al., 2008), Suzuki09 refers to the parser of Suzuki et al. (2009), Chen09-ord2s refers to the second-order parser with subtree-based features of Chen et al. (2009), and Zhou11 refers to the second-order parser with web-derived selectional preference features of Zhou et al. (2011).

The results showed that our MSTB-DLM2 obtained the comparable accuracy with the previous state-of-the-art systems. Koo10-model1 (Koo and Collins, 2010) used the third-order features and achieved the best reported result among the supervised parsers. Suzuki2009 (Suzuki et al., 2009) reported the best reported result by combining a Semi-supervised Structured Conditional Model (Suzuki and Isozaki, 2008) with the method of (Koo et al., 2008). However, their decoding complexities were higher than ours and we believe that the performance of our parser can be further enhanced by integrating their methods with our parser.

Type	System	UAS	Cost
G	McDonald06	91.5	$O(n^3)$
	Koo08-standard	92.02	$O(n^4)$
	Koo10-model1	93.04	$O(n^4)$
S	Koo08-dep2c	93.16	$O(n^4)$
	Suzuki09	93.79	$O(n^4)$
	Chen09-ord2s	92.51	$O(n^3)$
	Zhou11	92.64	$O(n^4)$
D	MSTB-DLM2	92.76	$O(Kn^3)$

Table 7: Relevant results for English. G denotes the supervised graph-based parsers, S denotes the graph-based parsers with semi-supervised methods, D denotes our new parsers

## 6.7 Compare with previous work on Chinese

Table 8 shows the comparative results, where Chen08 refers to the parser of (Chen et al., 2008), Yu08 refers to the parser of (Yu et al., 2008), Zhao09 refers to the parser of (Zhao et al., 2009), and Chen09-ord2s refers to the second-order parser with subtree-based features of Chen et al. (2009). The results showed that our score for this data was the

best reported so far and significantly higher than the previous scores.

System	UAS
Chen08	86.52
Yu08	87.26
Zhao09	87.0
Chen09-ord2s	89.43
MSTB-DLM2	91.59

Table 8: Relevant results for Chinese

## 7 Analysis

Dependency parsers tend to perform worse on heads which have many children. Here, we studied the effect of DLM-based features for this structure. We calculated the number of children for each head and listed the accuracy changes for different numbers. We compared the MST-DLM1 and MST1 systems on the English data. The accuracy is the percentage of heads having all the correct children.

Figure 5 shows the results for English, where the X-axis represents the number of children, the Y-axis represents the accuracies, OURS refers to MST-DLM1, and Baseline refers to MST1. For example, for heads having two children, Baseline obtained 89.04% accuracy while OURS obtained 89.32%. From the figure, we found that OURS achieved better performance consistently in all cases and when the larger the number of children became, the more significant the performance improvement was.

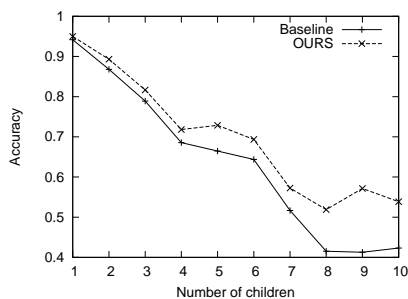


Figure 5: Improvement relative to numbers of children

## 8 Related work

Several previous studies related to our work have been conducted.

Koo et al. (2008) used a clustering algorithm to produce word clusters on a large amount of unannotated data and represented new features based on the clusters for dependency parsing models. Chen et al. (2009) proposed an approach that extracted partial tree structures from a large amount of data and used them as the additional features to improve dependency parsing. Their approaches were still restricted in a small number of arcs in the graphs. Suzuki et al. (2009) presented a semi-supervised learning approach. They extended a Semi-supervised Structured Conditional Model (SS-SCM)(Suzuki and Isozaki, 2008) to the dependency parsing problem and combined their method with the approach of Koo et al. (2008). In future work, we may consider apply their methods on our parsers to improve further.

Another group of methods are the co-training/self-training techniques. McClosky et al. (2006) presented a self-training approach for phrase structure parsing. Sagae and Tsujii (2007) used the co-training technique to improve performance. First, two parsers were used to parse the sentences in unannotated data. Then they selected some sentences which have the same trees produced by those two parsers. They retrained a parser on newly parsed sentences and the original labeled data. We are able to use the output of our systems for co-training/self-training techniques.

## 9 Conclusion

We have presented an approach to utilizing the dependency language model to improve graph-based dependency parsing. We represent new features based on the dependency language model and integrate them in the decoding algorithm directly using beam-search. Our approach enriches the feature representations but without increasing the decoding complexity. When tested on both English and Chinese data, our parsers provided very competitive performance compared with the best systems on the English data and achieved the best performance on the Chinese data in the literature.

## References

S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proc. of*



- CoNLL-X. SIGNLL.
- Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 957–961, Prague, Czech Republic, June. Association for Computational Linguistics.
- Eugene Charniak, Don Blaheta, Niyu Ge, Keith Hall, John Hale, and Mark Johnson. 2000. BLLIP 1987-89 WSJ Corpus Release 1, LDC2000T43. *Linguistic Data Consortium*.
- Wenliang Chen, Daisuke Kawahara, Kiyotaka Uchimoto, Yujie Zhang, and Hitoshi Isahara. 2008. Dependency parsing with short dependency relations in unlabeled data. In *Proceedings of IJCNLP 2008*.
- Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of EMNLP 2009*, pages 570–579, Singapore, August.
- Michael A. Covington. 2001. A fundamental algorithm for dependency parsing. In *Proceedings of the 39th Annual ACM Southeast Conference*, pages 95–102.
- Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3:951–991.
- J. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING1996*, pages 340–345.
- Chu-Ren Huang. 2009. Tagged Chinese Gigaword Version 2.0, LDC2009T14. *Linguistic Data Consortium*.
- Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of ACL 2010*, pages 1–11, Uppsala, Sweden, July. Association for Computational Linguistics.
- T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, June.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of ACL-IJCNLP2009*, pages 513–521, Suntec, Singapore, August. Association for Computational Linguistics.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- D. McClosky, E. Charniak, and M. Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of Coling-ACL*, pages 337–344.
- R. McDonald and J. Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of EMNLP-CoNLL*, pages 122–131.
- Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL 2006*, pages 81–88.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL 2005*, pages 91–98. Association for Computational Linguistics.
- J. Nivre, J. Hall, and J. Nilsson. 2004. Memory-based dependency parsing. In *Proc. of CoNLL 2004*, pages 49–56.
- 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 *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.
- Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP 1996*, pages 133–142.
- K. Sagae and J. Tsujii. 2007. Dependency parsing and domain adaptation with LR models and parser ensembles. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1044–1050.
- Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio, June. Association for Computational Linguistics.
- Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of ACL-08: HLT*, pages 665–673, Columbus, Ohio, June. Association for Computational Linguistics.
- Jun Suzuki, Hideki Isozaki, Xavier Carreras, and Michael Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of EMNLP2009*, pages 551–560, Singapore, August. Association for Computational Linguistics.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT 2003*, pages 195–206.
- K. Yu, D. Kawahara, and S. Kurohashi. 2008. Chinese dependency parsing with large scale automatically constructed case structures. In *Proceedings of Coling 2008*, pages 1049–1056, Manchester, UK, August.
- Y. Zhang and S. Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of EMNLP 2008*, pages 562–571, Honolulu, Hawaii, October.
- Hai Zhao, Yan Song, Chunyu Kit, and Guodong Zhou. 2009. Cross language dependency parsing us-

ing a bilingual lexicon. In *Proceedings of ACL-IJCNLP2009*, pages 55–63, Suntec, Singapore, August. Association for Computational Linguistics.

Guangyou Zhou, Jun Zhao, Kang Liu, and Li Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of ACL-HLT2011*, pages 1556–1565, Portland, Oregon, USA, June. Association for Computational Linguistics.

# Spectral Learning of Latent-Variable PCFGs

Shay B. Cohen<sup>1</sup>, Karl Stratos<sup>1</sup>, Michael Collins<sup>1</sup>, Dean P. Foster<sup>2</sup>, and Lyle Ungar<sup>3</sup>

<sup>1</sup>Dept. of Computer Science, Columbia University

<sup>2</sup>Dept. of Statistics/<sup>3</sup>Dept. of Computer and Information Science, University of Pennsylvania

{scohen,stratos,mcollins}@cs.columbia.edu, foster@wharton.upenn.edu, ungar@cis.upenn.edu

## Abstract

We introduce a spectral learning algorithm for latent-variable PCFGs (Petrov et al., 2006). Under a separability (singular value) condition, we prove that the method provides consistent parameter estimates.

## 1 Introduction

Statistical models with hidden or latent variables are of great importance in natural language processing, speech, and many other fields. The EM algorithm is a remarkably successful method for parameter estimation within these models: it is simple, it is often relatively efficient, and it has well understood formal properties. It does, however, have a major limitation: it has no guarantee of finding the global optimum of the likelihood function. From a theoretical perspective, this means that the EM algorithm is not guaranteed to give consistent parameter estimates. From a practical perspective, problems with local optima can be difficult to deal with.

Recent work has introduced polynomial-time learning algorithms (and consistent estimation methods) for two important cases of hidden-variable models: Gaussian mixture models (Dasgupta, 1999; Vempala and Wang, 2004) and hidden Markov models (Hsu et al., 2009). These algorithms use spectral methods: that is, algorithms based on eigenvector decompositions of linear systems, in particular singular value decomposition (SVD). In the general case, learning of HMMs or GMMs is intractable (e.g., see Terwijn, 2002). Spectral methods finesse the problem of intractability by assuming separability conditions. For example, the algorithm of Hsu et al. (2009) has a sample complexity that is polynomial in  $1/\sigma$ , where  $\sigma$  is the minimum singular value of an underlying decomposition. These methods are not susceptible to problems with local maxima, and give consistent parameter estimates.

In this paper we derive a spectral algorithm for learning of latent-variable PCFGs (L-PCFGs) (Petrov et al., 2006; Matsuzaki et al., 2005). Our

method involves a significant extension of the techniques from Hsu et al. (2009). L-PCFGs have been shown to be a very effective model for natural language parsing. Under a separation (singular value) condition, our algorithm provides consistent parameter estimates; this is in contrast with previous work, which has used the EM algorithm for parameter estimation, with the usual problems of local optima.

The parameter estimation algorithm (see figure 4) is simple and efficient. The first step is to take an SVD of the training examples, followed by a projection of the training examples down to a low-dimensional space. In a second step, empirical averages are calculated on the training example, followed by standard matrix operations. On test examples, simple (tensor-based) variants of the inside-outside algorithm (figures 2 and 3) can be used to calculate probabilities and marginals of interest.

Our method depends on the following results:

- *Tensor form of the inside-outside algorithm.* Section 5 shows that the inside-outside algorithm for L-PCFGs can be written using tensors. Theorem 1 gives conditions under which the tensor form calculates inside and outside terms correctly.

- *Observable representations.* Section 6 shows that under a singular-value condition, there is an *observable form* for the tensors required by the inside-outside algorithm. By an observable form, we follow the terminology of Hsu et al. (2009) in referring to quantities that can be estimated directly from data where values for latent variables are unobserved. Theorem 2 shows that tensors derived from the observable form satisfy the conditions of theorem 1.

- *Estimating the model.* Section 7 gives an algorithm for estimating parameters of the observable representation from training data. Theorem 3 gives a sample complexity result, showing that the estimates converge to the true distribution at a rate of  $1/\sqrt{M}$  where  $M$  is the number of training examples.

The algorithm is strikingly different from the EM algorithm for L-PCFGs, both in its basic form, and in its consistency guarantees. The techniques de-

veloped in this paper are quite general, and should be relevant to the development of spectral methods for estimation in other models in NLP, for example alignment models for translation, synchronous PCFGs, and so on. The tensor form of the inside-outside algorithm gives a new view of basic calculations in PCFGs, and may itself lead to new models.

## 2 Related Work

For work on L-PCFGs using the EM algorithm, see Petrov et al. (2006), Matsuzaki et al. (2005), Pereira and Schabes (1992). Our work builds on methods for learning of HMMs (Hsu et al., 2009; Foster et al., 2012; Jaeger, 2000), but involves several extensions: in particular in the tensor form of the inside-outside algorithm, and observable representations for the tensor form. Balle et al. (2011) consider spectral learning of finite-state transducers; Lague et al. (2012) considers spectral learning of head automata for dependency parsing. Parikh et al. (2011) consider spectral learning algorithms of tree-structured directed bayes nets.

## 3 Notation

Given a matrix  $A$  or a vector  $v$ , we write  $A^\top$  or  $v^\top$  for the associated transpose. For any integer  $n \geq 1$ , we use  $[n]$  to denote the set  $\{1, 2, \dots, n\}$ . For any row or column vector  $y \in \mathbb{R}^m$ , we use  $\text{diag}(y)$  to refer to the  $(m \times m)$  matrix with diagonal elements equal to  $y_h$  for  $h = 1 \dots m$ , and off-diagonal elements equal to 0. For any statement  $\Gamma$ , we use  $[[\Gamma]]$  to refer to the indicator function that is 1 if  $\Gamma$  is true, and 0 if  $\Gamma$  is false. For a random variable  $X$ , we use  $\mathbb{E}[X]$  to denote its expected value.

We will make (quite limited) use of tensors:

**Definition 1** A tensor  $C \in \mathbb{R}^{(m \times m \times m)}$  is a set of  $m^3$  parameters  $C_{i,j,k}$  for  $i, j, k \in [m]$ . Given a tensor  $C$ , and a vector  $y \in \mathbb{R}^m$ , we define  $C(y)$  to be the  $(m \times m)$  matrix with components  $[C(y)]_{i,j} = \sum_{k \in [m]} C_{i,j,k} y_k$ . Hence  $C$  can be interpreted as a function  $C : \mathbb{R}^m \rightarrow \mathbb{R}^{(m \times m)}$  that maps a vector  $y \in \mathbb{R}^m$  to a matrix  $C(y)$  of dimension  $(m \times m)$ .

In addition, we define the tensor  $C_* \in \mathbb{R}^{(m \times m \times m)}$  for any tensor  $C \in \mathbb{R}^{(m \times m \times m)}$  to have values

$$[C_*]_{i,j,k} = C_{k,j,i}$$

Finally, for vectors  $x, y, z \in \mathbb{R}^m$ ,  $xy^\top z^\top$  is the tensor  $D \in \mathbb{R}^{m \times m \times m}$  where  $D_{j,k,l} = x_j y_k z_l$  (this is analogous to the outer product:  $[xy^\top]_{j,k} = x_j y_k$ ).

## 4 L-PCFGs: Basic Definitions

This section gives a definition of the L-PCFG formalism used in this paper. An L-PCFG is a 5-tuple  $(\mathcal{N}, \mathcal{I}, \mathcal{P}, m, n)$  where:

- $\mathcal{N}$  is the set of non-terminal symbols in the grammar.  $\mathcal{I} \subset \mathcal{N}$  is a finite set of *in-terminals*.  $\mathcal{P} \subset \mathcal{N}$  is a finite set of *pre-terminals*. We assume that  $\mathcal{N} = \mathcal{I} \cup \mathcal{P}$ , and  $\mathcal{I} \cap \mathcal{P} = \emptyset$ . Hence we have partitioned the set of non-terminals into two subsets.

- $[m]$  is the set of possible hidden states.
- $[n]$  is the set of possible words.
- For all  $a \in \mathcal{I}$ ,  $b \in \mathcal{N}$ ,  $c \in \mathcal{N}$ ,  $h_1, h_2, h_3 \in [m]$ , we have a context-free rule  $a(h_1) \rightarrow b(h_2) c(h_3)$ .
- For all  $a \in \mathcal{P}$ ,  $h \in [m]$ ,  $x \in [n]$ , we have a context-free rule  $a(h) \rightarrow x$ .

Hence each in-terminal  $a \in \mathcal{I}$  is always the left-hand-side of a binary rule  $a \rightarrow b c$ ; and each pre-terminal  $a \in \mathcal{P}$  is always the left-hand-side of a rule  $a \rightarrow x$ . Assuming that the non-terminals in the grammar can be partitioned this way is relatively benign, and makes the estimation problem cleaner.

We define the set of possible “skeletal rules” as  $\mathcal{R} = \{a \rightarrow b c : a \in \mathcal{I}, b \in \mathcal{N}, c \in \mathcal{N}\}$ . The parameters of the model are as follows:

- For each  $a \rightarrow b c \in \mathcal{R}$ , and  $h \in [m]$ , we have a parameter  $q(a \rightarrow b c | h, a)$ . For each  $a \in \mathcal{P}$ ,  $x \in [n]$ , and  $h \in [m]$ , we have a parameter  $q(a \rightarrow x | h, a)$ . For each  $a \rightarrow b c \in \mathcal{R}$ , and  $h, h' \in [m]$ , we have parameters  $s(h' | h, a \rightarrow b c)$  and  $t(h' | h, a \rightarrow b c)$ .

These definitions give a PCFG, with rule probabilities

$$p(a(h_1) \rightarrow b(h_2) c(h_3) | a(h_1)) = q(a \rightarrow b c | h_1, a) \times s(h_2 | h_1, a \rightarrow b c) \times t(h_3 | h_1, a \rightarrow b c)$$

$$\text{and } p(a(h) \rightarrow x | a(h)) = q(a \rightarrow x | h, a).$$

In addition, for each  $a \in \mathcal{I}$ , for each  $h \in [m]$ , we have a parameter  $\pi(a, h)$  which is the probability of non-terminal  $a$  paired with hidden variable  $h$  being at the root of the tree.

An L-PCFG defines a distribution over parse trees as follows. A *skeletal tree* (s-tree) is a sequence of rules  $r_1 \dots r_N$  where each  $r_i$  is either of the form  $a \rightarrow b c$  or  $a \rightarrow x$ . The rule sequence forms a top-down, left-most derivation under a CFG with skeletal rules. See figure 1 for an example.

A *full tree* consists of an s-tree  $r_1 \dots r_N$ , together with values  $h_1 \dots h_N$ . Each  $h_i$  is the value for

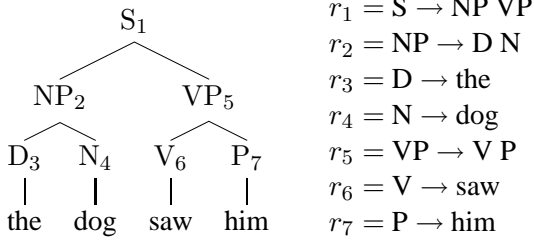


Figure 1: An s-tree, and its sequence of rules. (For convenience we have numbered the nodes in the tree.)

the hidden variable for the left-hand-side of rule  $r_i$ . Each  $h_i$  can take any value in  $[m]$ .

Define  $a_i$  to be the non-terminal on the left-hand-side of rule  $r_i$ . For any  $i \in \{2 \dots N\}$  define  $\text{pa}(i)$  to be the index of the rule above node  $i$  in the tree. Define  $L \subset [N]$  to be the set of nodes in the tree which are the left-child of some parent, and  $R \subset [N]$  to be the set of nodes which are the right-child of some parent. The probability mass function (PMF) over full trees is then

$$\begin{aligned}
 p(r_1 \dots r_N, h_1 \dots h_N) &= \pi(a_1, h_1) \\
 &\times \prod_{i=1}^N q(r_i | h_i, a_i) \times \prod_{i \in L} s(h_i | h_{\text{pa}(i)}, r_{\text{pa}(i)}) \\
 &\times \prod_{i \in R} t(h_i | h_{\text{pa}(i)}, r_{\text{pa}(i)}) \quad (1)
 \end{aligned}$$

The PMF over s-trees is  $p(r_1 \dots r_N) = \sum_{h_1 \dots h_N} p(r_1 \dots r_N, h_1 \dots h_N)$ .

In the remainder of this paper, we make use of matrix form of parameters of an L-PCFG, as follows:

- For each  $a \rightarrow b c \in \mathcal{R}$ , we define  $Q^{a \rightarrow b c} \in \mathbb{R}^{m \times m}$  to be the matrix with values  $q(a \rightarrow b c | h, a)$  for  $h = 1, 2, \dots, m$  on its diagonal, and 0 values for its off-diagonal elements. Similarly, for each  $a \in \mathcal{P}$ ,  $x \in [n]$ , we define  $Q^{a \rightarrow x} \in \mathbb{R}^{m \times m}$  to be the matrix with values  $q(a \rightarrow x | h, a)$  for  $h = 1, 2, \dots, m$  on its diagonal, and 0 values for its off-diagonal elements.
- For each  $a \rightarrow b c \in \mathcal{R}$ , we define  $S^{a \rightarrow b c} \in \mathbb{R}^{m \times m}$  where  $[S^{a \rightarrow b c}]_{h', h} = s(h' | h, a \rightarrow b c)$ .
- For each  $a \rightarrow b c \in \mathcal{R}$ , we define  $T^{a \rightarrow b c} \in \mathbb{R}^{m \times m}$  where  $[T^{a \rightarrow b c}]_{h', h} = t(h' | h, a \rightarrow b c)$ .
- For each  $a \in \mathcal{I}$ , we define the vector  $\pi^a \in \mathbb{R}^m$  where  $[\pi^a]_h = \pi(a, h)$ .

## 5 Tensor Form of the Inside-Outside Algorithm

Given an L-PCFG, two calculations are central:

**Inputs:** s-tree  $r_1 \dots r_N$ , L-PCFG  $(\mathcal{N}, \mathcal{I}, \mathcal{P}, m, n)$ , parameters

- $C^{a \rightarrow b c} \in \mathbb{R}^{(m \times m \times m)}$  for all  $a \rightarrow b c \in \mathcal{R}$
- $c_{a \rightarrow x}^\infty \in \mathbb{R}^{(1 \times m)}$  for all  $a \in \mathcal{P}, x \in [n]$
- $c_a^1 \in \mathbb{R}^{(m \times 1)}$  for all  $a \in \mathcal{I}$ .

**Algorithm:** (calculate the  $f^i$  terms bottom-up in the tree)

- For all  $i \in [N]$  such that  $a_i \in \mathcal{P}$ ,  $f^i = c_{r_i}^\infty$
- For all  $i \in [N]$  such that  $a_i \in \mathcal{I}$ ,  $f^i = f^\gamma C^{r_i}(f^\beta)$  where  $\beta$  is the index of the left child of node  $i$  in the tree, and  $\gamma$  is the index of the right child.

**Return:**  $f^1 c_{a_1}^1 = p(r_1 \dots r_N)$

Figure 2: The tensor form for calculation of  $p(r_1 \dots r_N)$ .

1. For a given s-tree  $r_1 \dots r_N$ , calculate  $p(r_1 \dots r_N)$ .
2. For a given input sentence  $x = x_1 \dots x_N$ , calculate the marginal probabilities

$$\mu(a, i, j) = \sum_{\tau \in \mathcal{T}(x): (a, i, j) \in \tau} p(\tau)$$

for each non-terminal  $a \in \mathcal{N}$ , for each  $(i, j)$  such that  $1 \leq i \leq j \leq N$ .

Here  $\mathcal{T}(x)$  denotes the set of all possible s-trees for the sentence  $x$ , and we write  $(a, i, j) \in \tau$  if non-terminal  $a$  spans words  $x_i \dots x_j$  in the parse tree  $\tau$ .

The marginal probabilities have a number of uses. Perhaps most importantly, for a given sentence  $x = x_1 \dots x_N$ , the parsing algorithm of Goodman (1996) can be used to find

$$\arg \max_{\tau \in \mathcal{T}(x)} \sum_{(a, i, j) \in \tau} \mu(a, i, j)$$

This is the parsing algorithm used by Petrov et al. (2006), for example. In addition, we can calculate the probability for an input sentence,  $p(x) = \sum_{\tau \in \mathcal{T}(x)} p(\tau)$ , as  $p(x) = \sum_{a \in \mathcal{I}} \mu(a, 1, N)$ .

Variants of the inside-outside algorithm can be used for problems 1 and 2. This section introduces a novel form of these algorithms, using tensors. This is the first step in deriving the spectral estimation method.

The algorithms are shown in figures 2 and 3. Each algorithm takes the following inputs:

1. A tensor  $C^{a \rightarrow b c} \in \mathbb{R}^{(m \times m \times m)}$  for each rule  $a \rightarrow b c$ .
2. A vector  $c_{a \rightarrow x}^\infty \in \mathbb{R}^{(1 \times m)}$  for each rule  $a \rightarrow x$ .

3. A vector  $c_a^1 \in \mathbb{R}^{(m \times 1)}$  for each  $a \in \mathcal{I}$ .

The following theorem gives conditions under which the algorithms are correct:

**Theorem 1** Assume that we have an L-PCFG with parameters  $Q^{a \rightarrow x}, Q^{a \rightarrow b c}, T^{a \rightarrow b c}, S^{a \rightarrow b c}, \pi^a$ , and that there exist matrices  $G^a \in \mathbb{R}^{(m \times m)}$  for all  $a \in \mathcal{N}$  such that each  $G^a$  is invertible, and such that:

1. For all rules  $a \rightarrow b c$ ,  $C^{a \rightarrow b c}(y) = G^c T^{a \rightarrow b c} \text{diag}(y G^b S^{a \rightarrow b c}) Q^{a \rightarrow b c} (G^a)^{-1}$
2. For all rules  $a \rightarrow x$ ,  $c_{a \rightarrow x}^\infty = 1^\top Q^{a \rightarrow x} (G^a)^{-1}$
3. For all  $a \in \mathcal{I}$ ,  $c_a^1 = G^a \pi^a$

Then: 1) The algorithm in figure 2 correctly computes  $p(r_1 \dots r_N)$  under the L-PCFG. 2) The algorithm in figure 3 correctly computes the marginals  $\mu(a, i, j)$  under the L-PCFG.

*Proof:* See section 9.1.  $\square$

## 6 Estimating the Tensor Model

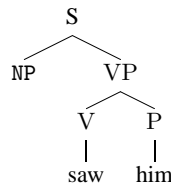
A crucial result is that it is possible to directly estimate parameters  $C^{a \rightarrow b c}$ ,  $c_{a \rightarrow x}^\infty$  and  $c_a^1$  that satisfy the conditions in theorem 1, from a training sample consisting of s-trees (i.e., trees where hidden variables are unobserved). We first describe random variables underlying the approach, then describe observable representations based on these random variables.

### 6.1 Random Variables Underlying the Approach

Each s-tree with  $N$  rules  $r_1 \dots r_N$  has  $N$  nodes. We will use the s-tree in figure 1 as a running example.

Each node has an associated rule: for example, node 2 in the tree in figure 1 has the rule  $\text{NP} \rightarrow \text{D N}$ . If the rule at a node is of the form  $a \rightarrow b c$ , then there are left and right *inside trees* below the left child and right child of the rule. For example, for node 2 we have a left inside tree rooted at node 3, and a right inside tree rooted at node 4 (in this case the left and right inside trees both contain only a single rule production, of the form  $a \rightarrow x$ ; however in the general case they might be arbitrary subtrees).

In addition, each node has an *outside tree*. For node 2, the outside tree is



**Inputs:** Sentence  $x_1 \dots x_N$ , L-PCFG  $(\mathcal{N}, \mathcal{I}, \mathcal{P}, m, n)$ , parameters  $C^{a \rightarrow b c} \in \mathbb{R}^{(m \times m \times m)}$  for all  $a \rightarrow b c \in \mathcal{R}$ ,  $c_{a \rightarrow x}^\infty \in \mathbb{R}^{(1 \times m)}$  for all  $a \in \mathcal{P}, x \in [n]$ ,  $c_a^1 \in \mathbb{R}^{(m \times 1)}$  for all  $a \in \mathcal{I}$ .

**Data structures:**

- Each  $\alpha^{a,i,j} \in \mathbb{R}^{1 \times m}$  for  $a \in \mathcal{N}, 1 \leq i \leq j \leq N$  is a row vector of inside terms.
- Each  $\beta^{a,i,j} \in \mathbb{R}^{m \times 1}$  for  $a \in \mathcal{N}, 1 \leq i \leq j \leq N$  is a column vector of outside terms.
- Each  $\mu(a, i, j) \in \mathbb{R}$  for  $a \in \mathcal{N}, 1 \leq i \leq j \leq N$  is a marginal probability.

**Algorithm:**

(Inside base case)  $\forall a \in \mathcal{P}, i \in [N], \alpha^{a,i,i} = c_{a \rightarrow x_i}^\infty$   
 (Inside recursion)  $\forall a \in \mathcal{I}, 1 \leq i < j \leq N,$

$$\alpha^{a,i,j} = \sum_{k=i}^{j-1} \sum_{a \rightarrow b c} \alpha^{c,k+1,j} C^{a \rightarrow b c} (\alpha^{b,i,k})$$

(Outside base case)  $\forall a \in \mathcal{I}, \beta^{a,1,n} = c_a^1$   
 (Outside recursion)  $\forall a \in \mathcal{N}, 1 \leq i \leq j \leq N,$

$$\beta^{a,i,j} = \sum_{k=1}^{i-1} \sum_{b \rightarrow c a} C^{b \rightarrow c a} (\alpha^{c,k,i-1}) \beta^{b,k,j}$$

$$+ \sum_{k=j+1}^N \sum_{b \rightarrow a c} C_*^{b \rightarrow a c} (\alpha^{c,j+1,k}) \beta^{b,i,k}$$

(Marginals)  $\forall a \in \mathcal{N}, 1 \leq i \leq j \leq N,$

$$\mu(a, i, j) = \alpha^{a,i,j} \beta^{a,i,j} = \sum_{h \in [m]} \alpha_h^{a,i,j} \beta_h^{a,i,j}$$

Figure 3: The tensor form of the inside-outside algorithm, for calculation of marginal terms  $\mu(a, i, j)$ .

The outside tree contains everything in the s-tree  $r_1 \dots r_N$ , excluding the subtree below node  $i$ .

Our random variables are defined as follows. First, we select a random internal node, from a random tree, as follows:

- Sample an s-tree  $r_1 \dots r_N$  from the PMF  $p(r_1 \dots r_N)$ . Choose a node  $i$  uniformly at random from  $[N]$ .

If the rule  $r_i$  for the node  $i$  is of the form  $a \rightarrow b c$ , we define random variables as follows:

- $R_1$  is equal to the rule  $r_i$  (e.g.,  $\text{NP} \rightarrow \text{D N}$ ).
- $T_1$  is the inside tree rooted at node  $i$ .  $T_2$  is the inside tree rooted at the left child of node  $i$ , and  $T_3$  is the inside tree rooted at the right child of node  $i$ .
- $H_1, H_2, H_3$  are the hidden variables associated with node  $i$ , the left child of node  $i$ , and the right child of node  $i$  respectively.

- $A_1, A_2, A_3$  are the labels for node  $i$ , the left child of node  $i$ , and the right child of node  $i$  respectively. (E.g.,  $A_1 = \text{NP}$ ,  $A_2 = \text{D}$ ,  $A_3 = \text{N}$ .)

- $O$  is the outside tree at node  $i$ .

- $B$  is equal to 1 if node  $i$  is at the root of the tree (i.e.,  $i = 1$ ), 0 otherwise.

If the rule  $r_i$  for the selected node  $i$  is of the form  $a \rightarrow x$ , we have random variables  $R_1, T_1, H_1, A_1, O, B$  as defined above, but  $H_2, H_3, T_2, T_3, A_2$ , and  $A_3$  are not defined.

We assume a function  $\psi$  that maps outside trees  $o$  to feature vectors  $\psi(o) \in \mathbb{R}^{d'}$ . For example, the feature vector might track the rule directly above the node in question, the word following the node in question, and so on. We also assume a function  $\phi$  that maps inside trees  $t$  to feature vectors  $\phi(t) \in \mathbb{R}^d$ . As one example, the function  $\phi$  might be an indicator function tracking the rule production at the root of the inside tree. Later we give formal criteria for what makes good definitions of  $\psi(o)$  or  $\phi(t)$ . One requirement is that  $d' \geq m$  and  $d \geq m$ .

In tandem with these definitions, we assume projection matrices  $U^a \in \mathbb{R}^{(d \times m)}$  and  $V^a \in \mathbb{R}^{(d' \times m)}$  for all  $a \in \mathcal{N}$ . We then define additional random variables  $Y_1, Y_2, Y_3, Z$  as

$$Y_1 = (U^{a_1})^\top \phi(T_1) \quad Z = (V^{a_1})^\top \psi(O)$$

$$Y_2 = (U^{a_2})^\top \phi(T_2) \quad Y_3 = (U^{a_3})^\top \phi(T_3)$$

where  $a_i$  is the value of the random variable  $A_i$ . Note that  $Y_1, Y_2, Y_3, Z$  are all in  $\mathbb{R}^m$ .

## 6.2 Observable Representations

Given the definitions in the previous section, our representation is based on the following matrix, tensor and vector quantities, defined for all  $a \in \mathcal{N}$ , for all rules of the form  $a \rightarrow b c$ , and for all rules of the form  $a \rightarrow x$  respectively:

$$\begin{aligned} \Sigma^a &= \mathbf{E}[Y_1 Z^\top | A_1 = a] \\ D^{a \rightarrow b c} &= \mathbf{E} \left[ [[R_1 = a \rightarrow b c]] Y_3 Z^\top Y_2^\top | A_1 = a \right] \\ d_{a \rightarrow x}^\infty &= \mathbf{E} \left[ [[R_1 = a \rightarrow x]] Z^\top | A_1 = a \right] \end{aligned}$$

Assuming access to functions  $\phi$  and  $\psi$ , and projection matrices  $U^a$  and  $V^a$ , these quantities can be estimated directly from training data consisting of a set of s-trees (see section 7).

Our observable representation then consists of:

$$C^{a \rightarrow b c}(y) = D^{a \rightarrow b c}(y) (\Sigma^a)^{-1} \quad (2)$$

$$c_{a \rightarrow x}^\infty = d_{a \rightarrow x}^\infty (\Sigma^a)^{-1} \quad (3)$$

$$c_a^1 = \mathbf{E} [[A_1 = a]] Y_1 | B = 1 \quad (4)$$

We next introduce conditions under which these quantities satisfy the conditions in theorem 1.

The following definition will be important:

**Definition 2** For all  $a \in \mathcal{N}$ , we define the matrices  $I^a \in \mathbb{R}^{(d \times m)}$  and  $J^a \in \mathbb{R}^{(d' \times m)}$  as

$$[I^a]_{i,h} = \mathbf{E}[\phi_i(T_1) | H_1 = h, A_1 = a]$$

$$[J^a]_{i,h} = \mathbf{E}[\psi_i(O) | H_1 = h, A_1 = a]$$

In addition, for any  $a \in \mathcal{N}$ , we use  $\gamma^a \in \mathbb{R}^m$  to denote the vector with  $\gamma_h^a = P(H_1 = h | A_1 = a)$ .

The correctness of the representation will rely on the following conditions being satisfied (these are parallel to conditions 1 and 2 in Hsu et al. (2009)):

**Condition 1**  $\forall a \in \mathcal{N}$ , the matrices  $I^a$  and  $J^a$  are of full rank (i.e., they have rank  $m$ ). For all  $a \in \mathcal{N}$ , for all  $h \in [m]$ ,  $\gamma_h^a > 0$ .

**Condition 2**  $\forall a \in \mathcal{N}$ , the matrices  $U^a \in \mathbb{R}^{(d \times m)}$  and  $V^a \in \mathbb{R}^{(d' \times m)}$  are such that the matrices  $G^a = (U^a)^\top I^a$  and  $K^a = (V^a)^\top J^a$  are invertible.

The following lemma justifies the use of an SVD calculation as one method for finding values for  $U^a$  and  $V^a$  that satisfy condition 2:

**Lemma 1** Assume that condition 1 holds, and for all  $a \in \mathcal{N}$  define

$$\Omega^a = \mathbf{E}[\phi(T_1) (\psi(O))^\top | A_1 = a] \quad (5)$$

Then if  $U^a$  is a matrix of the  $m$  left singular vectors of  $\Omega^a$  corresponding to non-zero singular values, and  $V^a$  is a matrix of the  $m$  right singular vectors of  $\Omega^a$  corresponding to non-zero singular values, then condition 2 is satisfied.

*Proof sketch:* It can be shown that  $\Omega^a = I^a \text{diag}(\gamma^a) (J^a)^\top$ . The remainder is similar to the proof of lemma 2 in Hsu et al. (2009).  $\square$

The matrices  $\Omega^a$  can be estimated directly from a training set consisting of s-trees, assuming that we have access to the functions  $\phi$  and  $\psi$ .

We can now state the following theorem:

**Theorem 2** Assume conditions 1 and 2 are satisfied. For all  $a \in \mathcal{N}$ , define  $G^a = (U^a)^\top I^a$ . Then under the definitions in Eqs. 2-4:

1. For all rules  $a \rightarrow b c$ ,  $C^{a \rightarrow b c}(y) = G^c T^{a \rightarrow b c} \text{diag}(y G^b S^{a \rightarrow b c}) Q^{a \rightarrow b c} (G^a)^{-1}$
2. For all rules  $a \rightarrow x$ ,  $c_{a \rightarrow x}^\infty = 1^\top Q^{a \rightarrow x} (G^a)^{-1}$ .
3. For all  $a \in \mathcal{N}$ ,  $c_a^1 = G^a \pi^a$

*Proof:* The following identities hold (see section 9.2):

$$D^{a \rightarrow b c}(y) = \quad (6)$$

$$G^c T^{a \rightarrow b c} \text{diag}(y G^b S^{a \rightarrow b c}) Q^{a \rightarrow b c} \text{diag}(\gamma^a) (K^a)^\top$$

$$d_{a \rightarrow x}^\infty = 1^\top Q^{a \rightarrow x} \text{diag}(\gamma^a) (K^a)^\top \quad (7)$$

$$\Sigma^a = G^a \text{diag}(\gamma^a) (K^a)^\top \quad (8)$$

$$c_a^1 = G^a \pi^a \quad (9)$$

Under conditions 1 and 2,  $\Sigma^a$  is invertible, and  $(\Sigma^a)^{-1} = ((K^a)^\top)^{-1} (\text{diag}(\gamma^a))^{-1} (G^a)^{-1}$ . The identities in the theorem follow immediately.  $\square$

## 7 Deriving Empirical Estimates

Figure 4 shows an algorithm that derives estimates of the quantities in Eqs 2, 3, and 4. As input, the algorithm takes a sequence of tuples  $(r^{(i,1)}, t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}, b^{(i)})$  for  $i \in [M]$ .

These tuples can be derived from a training set consisting of s-trees  $\tau_1 \dots \tau_M$  as follows:

- $\forall i \in [M]$ , choose a single node  $j_i$  uniformly at random from the nodes in  $\tau_i$ . Define  $r^{(i,1)}$  to be the rule at node  $j_i$ .  $t^{(i,1)}$  is the inside tree rooted at node  $j_i$ . If  $r^{(i,1)}$  is of the form  $a \rightarrow b c$ , then  $t^{(i,2)}$  is the inside tree under the left child of node  $j_i$ , and  $t^{(i,3)}$  is the inside tree under the right child of node  $j_i$ . If  $r^{(i,1)}$  is of the form  $a \rightarrow x$ , then  $t^{(i,2)} = t^{(i,3)} = \text{NULL}$ .  $o^{(i)}$  is the outside tree at node  $j_i$ .  $b^{(i)}$  is 1 if node  $j_i$  is at the root of the tree, 0 otherwise.

Under this process, assuming that the s-trees  $\tau_1 \dots \tau_M$  are i.i.d. draws from the distribution  $p(\tau)$  over s-trees under an L-PCFG, the tuples  $(r^{(i,1)}, t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}, b^{(i)})$  are i.i.d. draws from the joint distribution over the random variables  $R_1, T_1, T_2, T_3, O, B$  defined in the previous section.

The algorithm first computes estimates of the projection matrices  $U^a$  and  $V^a$ : following lemma 1, this is done by first deriving estimates of  $\Omega^a$ , and then taking SVDs of each  $\Omega^a$ . The matrices are then used to project inside and outside trees

$t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}$  down to  $m$ -dimensional vectors  $y^{(i,1)}, y^{(i,2)}, y^{(i,3)}, z^{(i)}$ ; these vectors are used to derive the estimates of  $C^{a \rightarrow b c}$ ,  $c_{a \rightarrow x}^\infty$ , and  $c_a^1$ .

We now state a PAC-style theorem for the learning algorithm. First, for a given L-PCFG, we need a couple of definitions:

- $\Lambda$  is the minimum absolute value of any element of the vectors/matrices/tensors  $c_a^1, d_{a \rightarrow x}^\infty, D^{a \rightarrow b c}, (\Sigma^a)^{-1}$ . (Note that  $\Lambda$  is a function of the projection matrices  $U^a$  and  $V^a$  as well as the underlying L-PCFG.)

- For each  $a \in \mathcal{N}$ ,  $\sigma^a$  is the value of the  $m$ 'th largest singular value of  $\Omega^a$ . Define  $\sigma = \min_a \sigma^a$ .

We then have the following theorem:

**Theorem 3** Assume that the inputs to the algorithm in figure 4 are i.i.d. draws from the joint distribution over the random variables  $R_1, T_1, T_2, T_3, O, B$ , under an L-PCFG with distribution  $p(r_1 \dots r_N)$  over s-trees. Define  $m$  to be the number of latent states in the L-PCFG. Assume that the algorithm in figure 4 has projection matrices  $\hat{U}^a$  and  $\hat{V}^a$  derived as left and right singular vectors of  $\Omega^a$ , as defined in Eq. 5. Assume that the L-PCFG, together with  $\hat{U}^a$  and  $\hat{V}^a$ , has coefficients  $\Lambda > 0$  and  $\sigma > 0$ . In addition, assume that all elements in  $c_a^1, d_{a \rightarrow x}^\infty, D^{a \rightarrow b c}$ , and  $\Sigma^a$  are in  $[-1, +1]$ . For any s-tree  $r_1 \dots r_N$  define  $\hat{p}(r_1 \dots r_N)$  to be the value calculated by the algorithm in figure 3 with inputs  $\hat{c}_a^1, \hat{d}_{a \rightarrow x}^\infty, \hat{C}^{a \rightarrow b c}$  derived from the algorithm in figure 4. Define  $R$  to be the total number of rules in the grammar of the form  $a \rightarrow b c$  or  $a \rightarrow x$ . Define  $M_a$  to be the number of training examples in the input to the algorithm in figure 4 where  $r^{i,1}$  has non-terminal  $a$  on its left-hand-side. Under these assumptions, if for all  $a$

$$M_a \geq \frac{128m^2}{(\sqrt{2N+1} + \epsilon - 1)^2 \Lambda^2 \sigma^4} \log \left( \frac{2mR}{\delta} \right)$$

Then

$$1 - \epsilon \leq \left| \frac{\hat{p}(r_1 \dots r_N)}{p(r_1 \dots r_N)} \right| \leq 1 + \epsilon$$

A similar theorem (omitted for space) states that  $1 - \epsilon \leq \left| \frac{\hat{\mu}(a,i,j)}{\mu(a,i,j)} \right| \leq 1 + \epsilon$  for the marginals.

The condition that  $\hat{U}^a$  and  $\hat{V}^a$  are derived from  $\Omega^a$ , as opposed to the sample estimate  $\hat{\Omega}^a$ , follows Foster et al. (2012). As these authors note, similar techniques to those of Hsu et al. (2009) should be



applicable in deriving results for the case where  $\hat{\Omega}^a$  is used in place of  $\Omega^a$ .

*Proof sketch:* The proof is similar to that of Foster et al. (2012). The basic idea is to first show that under the assumptions of the theorem, the estimates  $\hat{c}_a^1$ ,  $\hat{d}_{a \rightarrow x}^\infty$ ,  $\hat{D}^{a \rightarrow b c}$ ,  $\hat{\Sigma}^a$  are all close to the underlying values being estimated. The second step is to show that this ensures that  $\frac{\hat{p}(r_1 \dots r_{N'})}{p(r_1 \dots r_{N'})}$  is close to 1.  $\square$

The method described of selecting a single tuple  $(r^{(i,1)}, t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}, b^{(i)})$  for each s-tree ensures that the samples are i.i.d., and simplifies the analysis underlying theorem 3. In practice, an implementation should most likely use all nodes in all trees in training data; by Rao-Blackwellization we know such an algorithm would be better than the one presented, but the analysis of how much better would be challenging. It would almost certainly lead to a faster rate of convergence of  $\hat{p}$  to  $p$ .

## 8 Discussion

There are several potential applications of the method. The most obvious is parsing with L-PCFGs.<sup>1</sup> The approach should be applicable in other cases where EM has traditionally been used, for example in semi-supervised learning. Latent-variable HMMs for sequence labeling can be derived as special case of our approach, by converting tagged sequences to right-branching skeletal trees.

The sample complexity of the method depends on the minimum singular values of  $\Omega^a$ ; these singular values are a measure of how well correlated  $\psi$  and  $\phi$  are with the unobserved hidden variable  $H_1$ . Experimental work is required to find a good choice of values for  $\psi$  and  $\phi$  for parsing.

## 9 Proofs

This section gives proofs of theorems 1 and 2. Due to space limitations we cannot give full proofs; instead we provide proofs of some key lemmas. A long version of this paper will give the full proofs.

### 9.1 Proof of Theorem 1

First, the following lemma leads directly to the correctness of the algorithm in figure 2:

<sup>1</sup>Parameters can be estimated using the algorithm in figure 4; for a test sentence  $x_1 \dots x_N$  we can first use the algorithm in figure 3 to calculate marginals  $\mu(a, i, j)$ , then use the algorithm of Goodman (1996) to find  $\arg \max_{\tau \in \mathcal{T}(x)} \sum_{(a,i,j) \in \tau} \mu(a, i, j)$ .

**Inputs:** Training examples  $(r^{(i,1)}, t^{(i,1)}, t^{(i,2)}, t^{(i,3)}, o^{(i)}, b^{(i)})$  for  $i \in \{1 \dots M\}$ , where  $r^{(i,1)}$  is a context free rule;  $t^{(i,1)}$ ,  $t^{(i,2)}$  and  $t^{(i,3)}$  are inside trees;  $o^{(i)}$  is an outside tree; and  $b^{(i)} = 1$  if the rule is at the root of tree, 0 otherwise. A function  $\phi$  that maps inside trees  $t$  to feature-vectors  $\phi(t) \in \mathbb{R}^d$ . A function  $\psi$  that maps outside trees  $o$  to feature-vectors  $\psi(o) \in \mathbb{R}^{d'}$ .

**Algorithm:**

Define  $a_i$  to be the non-terminal on the left-hand side of rule  $r^{(i,1)}$ . If  $r^{(i,1)}$  is of the form  $a \rightarrow b c$ , define  $b_i$  to be the non-terminal for the left-child of  $r^{(i,1)}$ , and  $c_i$  to be the non-terminal for the right-child.

(Step 0: Singular Value Decompositions)

- Use the algorithm in figure 5 to calculate matrices  $\hat{U}^a \in \mathbb{R}^{(d \times m)}$  and  $\hat{V}^a \in \mathbb{R}^{(d' \times m)}$  for each  $a \in \mathcal{N}$ .

(Step 1: Projection)

- For all  $i \in [M]$ , compute  $y^{(i,1)} = (\hat{U}^{a_i})^\top \phi(t^{(i,1)})$ .
- For all  $i \in [M]$  such that  $r^{(i,1)}$  is of the form  $a \rightarrow b c$ , compute  $y^{(i,2)} = (\hat{U}^{b_i})^\top \phi(t^{(i,2)})$  and  $y^{(i,3)} = (\hat{U}^{c_i})^\top \phi(t^{(i,3)})$ .
- For all  $i \in [M]$ , compute  $z^{(i)} = (\hat{V}^{a_i})^\top \psi(o^{(i)})$ .

(Step 2: Calculate Correlations)

- For each  $a \in \mathcal{N}$ , define  $\delta_a = 1 / \sum_{i=1}^M [[a_i = a]]$
- For each rule  $a \rightarrow b c$ , compute  $\hat{D}^{a \rightarrow b c} = \delta_a \times \sum_{i=1}^M [[r^{(i,1)} = a \rightarrow b c]] y^{(i,2)} (z^{(i)})^\top (y^{(i,1)})^\top$
- For each rule  $a \rightarrow x$ , compute  $\hat{d}_{a \rightarrow x}^\infty = \delta_a \times \sum_{i=1}^M [[r^{(i,1)} = a \rightarrow x]] (z^{(i)})^\top$
- For each  $a \in \mathcal{N}$ , compute  $\hat{\Sigma}^a = \delta_a \times \sum_{i=1}^M [[a_i = a]] y^{(i,1)} (z^{(i)})^\top$

(Step 3: Compute Final Parameters)

- For all  $a \rightarrow b c$ ,  $\hat{C}^{a \rightarrow b c}(y) = \hat{D}^{a \rightarrow b c}(y) (\hat{\Sigma}^a)^{-1}$
- For all  $a \rightarrow x$ ,  $\hat{c}_{a \rightarrow x}^\infty = \hat{d}_{a \rightarrow x}^\infty (\hat{\Sigma}^a)^{-1}$
- For all  $a \in \mathcal{I}$ ,  $\hat{c}_a^1 = \frac{\sum_{i=1}^M [[a_i = a \text{ and } b^{(i)} = 1]] y^{(i,1)}}{\sum_{i=1}^M [[b^{(i)} = 1]]}$

Figure 4: The spectral learning algorithm.

**Inputs:** Identical to algorithm in figure 4.

**Algorithm:**

- For each  $a \in \mathcal{N}$ , compute  $\hat{\Omega}^a \in \mathbb{R}^{(d' \times d)}$  as

$$\hat{\Omega}^a = \frac{\sum_{i=1}^M [[a_i = a]] \phi(t^{(i,1)}) (\psi(o^{(i)}))^\top}{\sum_{i=1}^M [[a_i = a]]}$$

and calculate a singular value decomposition of  $\hat{\Omega}^a$ .

- For each  $a \in \mathcal{N}$ , define  $\hat{U}^a \in \mathbb{R}^{m \times d}$  to be a matrix of the left singular vectors of  $\hat{\Omega}^a$  corresponding to the  $m$  largest singular values. Define  $\hat{V}^a \in \mathbb{R}^{m \times d'}$  to be a matrix of the right singular vectors of  $\hat{\Omega}^a$  corresponding to the  $m$  largest singular values.

Figure 5: Singular value decompositions.

**Lemma 2** Assume that conditions 1-3 of theorem 1 are satisfied, and that the input to the algorithm in figure 2 is an  $s$ -tree  $r_1 \dots r_N$ . Define  $a_i$  for  $i \in [N]$  to be the non-terminal on the left-hand-side of rule  $r_i$ , and  $t_i$  for  $i \in [N]$  to be the  $s$ -tree with rule  $r_i$  at its root. Finally, for all  $i \in [N]$ , define the row vector  $b^i \in \mathbb{R}^{(1 \times m)}$  to have components

$$b_h^i = P(T_i = t_i | H_i = h, A_i = a_i)$$

for  $h \in [m]$ . Then for all  $i \in [N]$ ,  $f^i = b^i (G^{(a_i)})^{-1}$ . It follows immediately that

$$f^1 c_{a_1}^1 = b^1 (G^{(a_1)})^{-1} G^{a_1} \pi_{a_1} = p(r_1 \dots r_N) \quad \square$$

This lemma shows a direct link between the vectors  $f^i$  calculated in the algorithm, and the terms  $b_h^i$ , which are terms calculated by the conventional inside algorithm: each  $f^i$  is a linear transformation (through  $G^{(a_i)}$ ) of the corresponding vector  $b^i$ .

*Proof:* The proof is by induction.

First consider the base case. For any leaf—i.e., for any  $i$  such that  $a_i \in \mathcal{P}$ —we have  $b_h^i = q(r_i | h, a_i)$ , and it is easily verified that  $f^i = b^i (G^{(a_i)})^{-1}$ .

The inductive case is as follows. For all  $i \in [N]$  such that  $a_i \in \mathcal{I}$ , by the definition in the algorithm,

$$\begin{aligned} f^i &= f^\gamma C^{r_i} (f^\beta) \\ &= f^\gamma G^{a_\gamma} T^{r_i} \text{diag}(f^\beta G^{a_\beta} S^{r_i}) Q^{r_i} (G^{a_i})^{-1} \end{aligned}$$

Assuming by induction that  $f^\gamma = b^\gamma (G^{(a_\gamma)})^{-1}$  and  $f^\beta = b^\beta (G^{(a_\beta)})^{-1}$ , this simplifies to

$$f^i = \kappa^r \text{diag}(\kappa^l) Q^{r_i} (G^{a_i})^{-1} \quad (10)$$

where  $\kappa^r = b^\gamma T^{r_i}$ , and  $\kappa^l = b^\beta S^{r_i}$ .  $\kappa^r$  is a row vector with components  $\kappa_h^r = \sum_{h' \in [m]} b_{h'}^\gamma T_{h',h}^{r_i} = \sum_{h' \in [m]} b_{h'}^\gamma t(h' | h, r_i)$ . Similarly,  $\kappa^l$  is a row vector with components equal to  $\kappa_h^l = \sum_{h' \in [m]} b_{h'}^\beta S_{h',h}^{r_i} = \sum_{h' \in [m]} b_{h'}^\beta s(h' | h, r_i)$ . It can then be verified that  $\kappa^r \text{diag}(\kappa^l) Q^{r_i}$  is a row vector with components equal to  $\kappa_h^r \kappa_h^l q(r_i | h, a_i)$ .

But  $b_h^i = q(r_i | h, a_i) \times \left( \sum_{h' \in [m]} b_{h'}^\gamma t(h' | h, r_i) \right) \times \left( \sum_{h' \in [m]} b_{h'}^\beta s(h' | h, r_i) \right) = q(r_i | h, a_i) \kappa_h^r \kappa_h^l$ , hence  $\kappa^r \text{diag}(\kappa^l) Q^{r_i} = b^i$  and the inductive case follows immediately from Eq. 10.  $\square$

Next, we give a similar lemma, which implies the correctness of the algorithm in figure 3:

**Lemma 3** Assume that conditions 1-3 of theorem 1 are satisfied, and that the input to the algorithm in figure 3 is a sentence  $x_1 \dots x_N$ . For any  $a \in \mathcal{N}$ , for any  $1 \leq i \leq j \leq N$ , define  $\bar{\alpha}^{a,i,j} \in \mathbb{R}^{(1 \times m)}$  to have components  $\bar{\alpha}_h^{a,i,j} = p(x_i \dots x_j | h, a)$  for  $h \in [m]$ . In addition, define  $\bar{\beta}^{a,i,j} \in \mathbb{R}^{(m \times 1)}$  to have components  $\bar{\beta}_h^{a,i,j} = p(x_1 \dots x_{i-1}, a(h), x_{j+1} \dots x_N)$  for  $h \in [m]$ . Then for all  $i \in [N]$ ,  $\alpha^{a,i,j} = \bar{\alpha}^{a,i,j} (G^a)^{-1}$  and  $\beta^{a,i,j} = G^a \bar{\beta}^{a,i,j}$ . It follows that for all  $(a, i, j)$ ,

$$\begin{aligned} \mu(a, i, j) &= \bar{\alpha}^{a,i,j} (G^a)^{-1} G^a \bar{\beta}^{a,i,j} = \bar{\alpha}^{a,i,j} \bar{\beta}^{a,i,j} \\ &= \sum_h \bar{\alpha}_h^{a,i,j} \bar{\beta}_h^{a,i,j} = \sum_{\tau \in \mathcal{T}(x): (a,i,j) \in \tau} p(\tau) \quad \square \end{aligned}$$

Thus the vectors  $\alpha^{a,i,j}$  and  $\beta^{a,i,j}$  are linearly related to the vectors  $\bar{\alpha}^{a,i,j}$  and  $\bar{\beta}^{a,i,j}$ , which are the inside and outside terms calculated by the conventional form of the inside-outside algorithm.

The proof is by induction, and is similar to the proof of lemma 2; for reasons of space it is omitted.

## 9.2 Proof of the Identity in Eq. 6

We now prove the identity in Eq. 6, used in the proof of theorem 2. For reasons of space, we do not give the proofs of identities 7-9: the proofs are similar.

The following identities can be verified:

$$\begin{aligned} P(R_1 = a \rightarrow b \ c | H_1 = h, A_1 = a) &= q(a \rightarrow b \ c | h, a) \\ \mathbf{E}[Y_{3,j} | H_1 = h, R_1 = a \rightarrow b \ c] &= E_{j,h}^{a \rightarrow b \ c} \\ \mathbf{E}[Z_k | H_1 = h, R_1 = a \rightarrow b \ c] &= K_{k,h}^a \\ \mathbf{E}[Y_{2,l} | H_1 = h, R_1 = a \rightarrow b \ c] &= F_{l,h}^{a \rightarrow b \ c} \end{aligned}$$

where  $E^{a \rightarrow b \ c} = G^c T^{a \rightarrow b \ c}$ ,  $F^{a \rightarrow b \ c} = G^b S^{a \rightarrow b \ c}$ .

$Y_3$ ,  $Z$  and  $Y_2$  are independent when conditioned on  $H_1, R_1$  (this follows from the independence assumptions in the L-PCFG), hence

$$\begin{aligned} \mathbf{E}[[[R_1 = a \rightarrow b \ c]] Y_{3,j} Z_k Y_{2,l} | H_1 = h, A_1 = a] \\ = q(a \rightarrow b \ c | h, a) E_{j,h}^{a \rightarrow b \ c} K_{k,h}^a F_{l,h}^{a \rightarrow b \ c} \end{aligned}$$

Hence (recall that  $\gamma_h^a = P(H_1 = h | A_1 = a)$ ),

$$\begin{aligned} D_{j,k,l}^{a \rightarrow b \ c} &= \mathbf{E}[[[R_1 = a \rightarrow b \ c]] Y_{3,j} Z_k Y_{2,l} | A_1 = a] \\ &= \sum_h \gamma_h^a \mathbf{E}[[[R_1 = a \rightarrow b \ c]] Y_{3,j} Z_k Y_{2,l} | H_1 = h, A_1 = a] \\ &= \sum_h \gamma_h^a q(a \rightarrow b \ c | h, a) E_{j,h}^{a \rightarrow b \ c} K_{k,h}^a F_{l,h}^{a \rightarrow b \ c} \quad (11) \end{aligned}$$

from which Eq. 6 follows.  $\square$

**Acknowledgements:** Columbia University gratefully acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, AFRL, or the US government. Shay Cohen was supported by the National Science Foundation under Grant #1136996 to the Computing Research Association for the CIFellows Project. Dean Foster was supported by National Science Foundation grant 1106743.

## References

- B. Balle, A. Quattoni, and X. Carreras. 2011. A spectral learning algorithm for finite state transducers. In *Proceedings of ECML*.
- S. Dasgupta. 1999. Learning mixtures of Gaussians. In *Proceedings of FOCS*.
- Dean P. Foster, Jordan Rodu, and Lyle H. Ungar. 2012. Spectral dimensionality reduction for hmms. arXiv:1203.6130v1.
- J. Goodman. 1996. Parsing algorithms and metrics. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 177–183. Association for Computational Linguistics.
- D. Hsu, S. M. Kakade, and T. Zhang. 2009. A spectral algorithm for learning hidden Markov models. In *Proceedings of COLT*.
- H. Jaeger. 2000. Observable operator models for discrete stochastic time series. *Neural Computation*, 12(6).
- F. M. Lague, A. Quattoni, B. Balle, and X. Carreras. 2012. Spectral learning for non-deterministic dependency parsing. In *Proceedings of EACL*.
- T. Matsuzaki, Y. Miyao, and J. Tsujii. 2005. Probabilistic CFG with latent annotations. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 75–82. Association for Computational Linguistics.
- A. Parikh, L. Song, and E. P. Xing. 2011. A spectral algorithm for latent tree graphical models. In *Proceedings of The 28th International Conference on Machine Learning (ICML 2011)*.
- F. Pereira and Y. Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 128–135, Newark, Delaware, USA, June. Association for Computational Linguistics.
- S. Petrov, L. Barrett, R. Thibaux, and D. Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July. Association for Computational Linguistics.
- S. A. Terwijn. 2002. On the learnability of hidden markov models. In *Grammatical Inference: Algorithms and Applications (Amsterdam, 2002)*, volume 2484 of *Lecture Notes in Artificial Intelligence*, pages 261–268, Berlin. Springer.
- S. Vempala and G. Wang. 2004. A spectral algorithm for learning mixtures of distributions. *Journal of Computer and System Sciences*, 68(4):841–860.

# Reducing Approximation and Estimation Errors for Chinese Lexical Processing with Heterogeneous Annotations

Weiwei Sun<sup>†</sup> and Xiaojun Wan<sup>‡</sup> \*

<sup>†‡</sup>Institute of Computer Science and Technology, Peking University

<sup>†</sup>Saarbrücken Graduate School of Computer Science

<sup>†</sup>Department of Computational Linguistics, Saarland University

<sup>†</sup>Language Technology Lab, DFKI GmbH

{ws, wanxiaojun}@pku.edu.cn

## Abstract

We address the issue of consuming heterogeneous annotation data for Chinese word segmentation and part-of-speech tagging. We empirically analyze the diversity between two representative corpora, i.e. Penn Chinese Treebank (CTB) and PKU's People's Daily (PPD), on manually mapped data, and show that their linguistic annotations are systematically different and highly compatible. The analysis is further exploited to improve processing accuracy by (1) integrating systems that are respectively trained on heterogeneous annotations to reduce the approximation error, and (2) re-training models with high quality automatically converted data to reduce the estimation error. Evaluation on the CTB and PPD data shows that our novel model achieves a relative error reduction of 11% over the best reported result in the literature.

## 1 Introduction

A majority of data-driven NLP systems rely on large-scale, manually annotated corpora that are important to train statistical models but very expensive to build. Nowadays, for many tasks, multiple heterogeneous annotated corpora have been built and publicly available. For example, the Penn Treebank is popular to train PCFG-based parsers, while the Redwoods Treebank is well known for HPSG research; the Propbank is favored to build general semantic role labeling systems, while the FrameNet is attractive for predicate-specific labeling. The anno-

This work is mainly finished when the first author was in Saarland University and DFKI. Both authors are the corresponding authors.

tation schemes in different projects are usually different, since the underlying linguistic theories vary and have different ways to explain the same language phenomena. Though statistical NLP systems usually are not bound to specific annotation standards, almost all of them assume homogeneous annotation in the training corpus. The co-existence of heterogeneous annotation data therefore presents a new challenge to the consumers of such resources.

There are two essential characteristics of heterogeneous annotations that can be utilized to reduce two main types of errors in statistical NLP, i.e. the approximation error that is due to the intrinsic suboptimality of a model and the estimation error that is due to having only finite training data. First, heterogeneous annotations are (similar but) *different* as a result of different annotation schemata. Systems respectively trained on heterogeneous annotation data can produce different but relevant linguistic analysis. This suggests that complementary features from heterogeneous analysis can be derived for disambiguation, and therefore the approximation error can be reduced. Second, heterogeneous annotations are (different but) *similar* because their linguistic analysis is highly correlated. This implies that appropriate conversions between heterogeneous corpora could be reasonably accurate, and therefore the estimation error can be reduced by reason of the increase of reliable training data.

This paper explores heterogeneous annotations to reduce both approximation and estimation errors for Chinese word segmentation and part-of-speech (POS) tagging, which are fundamental steps for more advanced Chinese language processing tasks. We empirically analyze the diversity between two representative popular heterogeneous corpora, i.e.

Penn Chinese Treebank (CTB) and PKU’s People’s Daily (PPD). To that end, we manually label 200 sentences from CTB with PPD-style annotations.<sup>1</sup> Our analysis confirms the aforementioned two properties of heterogeneous annotations. Inspired by the sub-word tagging method introduced in (Sun, 2011), we propose a structure-based stacking model to fully utilize heterogeneous word structures to reduce the approximation error. In particular, joint word segmentation and POS tagging is addressed as a two step process. First, character-based taggers are respectively trained on heterogeneous annotations to produce multiple analysis. The outputs of these taggers are then merged into sub-word sequences, which are further re-segmented and tagged by a sub-word tagger. The sub-word tagger is designed to refine the tagging result with the help of heterogeneous annotations. To reduce the estimation error, we employ a learning-based approach to convert complementary heterogeneous data to increase labeled training data for the target task. Both the character-based tagger and the sub-word tagger can be refined by re-training with automatically converted data.

We conduct experiments on the CTB and PPD data, and compare our system with state-of-the-art systems. Our structure-based stacking model achieves an f-score of 94.36, which is superior to a feature-based stacking model introduced in (Jiang et al., 2009). The converted data can also enhance the baseline model. A simple character-based model can be improved from 93.41 to 94.11. Since the two treatments are concerned with reducing different types of errors and thus not fully overlapping, the combination of them gives a further improvement. Our final system achieves an f-score of 94.68, which yields a relative error reduction of 11% over the best published result (94.02).

## 2 Joint Chinese Word Segmentation and POS Tagging

Different from English and other Western languages, Chinese is written without explicit word delimiters such as space characters. To find and classify the

---

<sup>1</sup>The first 200 sentences of the development data for experiments are selected. This data set is submitted as a supplemental material for research purposes.

basic language units, i.e. words, word segmentation and POS tagging are important initial steps for Chinese language processing. Supervised learning with specifically defined training data has become a dominant paradigm. Joint approaches that resolve the two tasks simultaneously have received much attention in recent research. Previous work has shown that joint solutions led to accuracy improvements over pipelined systems by avoiding segmentation error propagation and exploiting POS information to help segmentation (Ng and Low, 2004; Jiang et al., 2008a; Zhang and Clark, 2008; Sun, 2011).

Two kinds of approaches are popular for joint word segmentation and POS tagging. The first is the “character-based” approach, where basic processing units are characters which compose words (Jiang et al., 2008a). In this kind of approach, the task is formulated as the classification of characters into POS tags with boundary information. For example, the label *B-NN* indicates that a character is located at the begging of a noun. Using this method, POS information is allowed to interact with segmentation. The second kind of solution is the “word-based” method, also known as semi-Markov tagging (Zhang and Clark, 2008; Zhang and Clark, 2010), where the basic predicting units are words themselves. This kind of solver sequentially decides whether the local sequence of characters makes up a word as well as its possible POS tag. Solvers may use previously predicted words and their POS information as clues to process a new word.

In addition, we proposed an effective and efficient stacked sub-word tagging model, which combines strengths of both character-based and word-based approaches (Sun, 2011). First, different character-based and word-based models are trained to produce multiple segmentation and tagging results. Second, the outputs of these coarse-grained models are merged into sub-word sequences, which are further bracketed and labeled with POS tags by a fine-grained sub-word tagger. Their solution can be viewed as utilizing stacked learning to integrate heterogeneous models.

Supervised segmentation and tagging can be improved by exploiting rich linguistic resources. Jiang et al. (2009) presented a preliminary study for annotation ensemble, which motivates our research as well as similar investigations for other NLP tasks,

e.g. parsing (Niu et al., 2009; Sun et al., 2010). In their solution, heterogeneous data is used to train an auxiliary segmentation and tagging system to produce informative features for target prediction. Our previous work (Sun and Xu, 2011) and Wang et al. (2011) explored unlabeled data to enhance strong supervised segmenters and taggers. Both of their work fall into the category of feature induction based semi-supervised learning. In brief, their methods harvest useful string knowledge from unlabeled or automatically analyzed data, and apply the knowledge to design new features for discriminative learning.

### 3 About Heterogeneous Annotations

For Chinese word segmentation and POS tagging, supervised learning has become a dominant paradigm. Much of the progress is due to the development of both corpora and machine learning techniques. Although several institutions to date have released their segmented and POS tagged data, acquiring sufficient quantities of high quality training examples is still a major bottleneck. The annotation schemes of existing lexical resources are different, since the underlying linguistic theories vary. Despite the existence of multiple resources, such data cannot be simply put together for training systems, because almost all of statistical NLP systems assume homogeneous annotation. Therefore, it is not only interesting but also important to study how to fully utilize heterogeneous resources to improve Chinese lexical processing.

There are two main types of errors in statistical NLP: (1) the approximation error that is due to the intrinsic suboptimality of a model and (2) the estimation error that is due to having only finite training data. Take Chinese word segmentation for example. Our previous analysis (Sun, 2010) shows that one main intrinsic disadvantage of character-based model is the difficulty in incorporating the whole word information, while one main disadvantage of word-based model is the weak ability to express word formation. In both models, the significant decrease of the prediction accuracy of out-of-vocabulary (OOV) words indicates the impact of the estimation error. The two essential characteristics about systematic diversity of heterogeneous annota-

tions can be utilized to reduce both approximation and estimation errors.

#### 3.1 Analysis of the CTB and PPD Standards

This paper focuses on two representative popular corpora for Chinese lexical processing: (1) the Penn Chinese Treebank (CTB) and (2) the PKU's People's Daily data (PPD). To analyze the diversity between their annotation standards, we pick up 200 sentences from CTB and manually label them according to the PPD standard. Specially, we employ a PPD-style segmentation and tagging system to automatically label these 200 sentences. A linguistic expert who deeply understands the PPD standard then manually checks the automatic analysis and corrects its errors.

These 200 sentences are segmented as 3886 and 3882 words respectively according to the CTB and PPD standards. The average lengths of word tokens are almost the same. However, the word boundaries or the definitions of words are different. 3561 word tokens are consistently segmented by both standards. In other words, 91.7% CTB word tokens share the same word boundaries with 91.6% PPD word tokens. Among these 3561 words, there are 552 punctuations that are simply consistently segmented. If punctuations are filtered out to avoid overestimation of consistency, 90.4% CTB words have same boundaries with 90.3% PPD words. The boundaries of words that are differently segmented are *compatible*. Among all annotations, only one cross-bracketing occurs. The statistics indicates that the two heterogeneous segmented corpora are systematically different, and confirms the aforementioned two properties of heterogeneous annotations.

Table 1 is the mapping between CTB-style tags and PPD-style tags. For the definition and illustration of these tags, please refers to the annotation guidelines<sup>2</sup>. The statistics after colons are how many times this POS tag pair appears among the 3561 words that are consistently segmented. From this table, we can see that (1) there is no one-to-one mapping between their heterogeneous word classification but (2) the mapping between heterogeneous tags is not very uncertain. This simple analysis indicates

<sup>2</sup>Available at <http://www.cis.upenn.edu/~chinese/posguide.3rd.ch.pdf> and [http://www.icl.pku.edu.cn/icl\\_groups/corpus/spec.htm](http://www.icl.pku.edu.cn/icl_groups/corpus/spec.htm).

that the two POS tagged corpora also hold the two properties of heterogeneous annotations. The differences between the POS annotation standards are systematic. The annotations in CTB are treebank-driven, and thus consider more functional (dynamic) information of basic lexical categories. The annotations in PPD are lexicon-driven, and thus focus on more *static* properties of words. Limited to the document length, we only illustrate the annotation of verbs and nouns for better understanding of the differences.

- The CTB tag **VV** indicates common verbs that are mainly labeled as verbs (v) too according to the PPD standard. However, these words can be also tagged as nominal categories (a, vn, n). The main reason is that there are a large number of Chinese adjectives and nouns that can be realized as predicates without linking verbs.
- The tag **NN** indicates common nouns in CTB. Some of them are labeled as verbal categories (vn, v). The main reason is that a majority of Chinese *verbs* could be realized as subjects and objects without form changes.

## 4 Structure-based Stacking

### 4.1 Reducing the Approximation Error via Stacking

Each annotation data set alone can yield a predictor that can be taken as a mechanism to produce structured texts. With different training data, we can construct multiple heterogeneous systems. These systems produce similar linguistic analysis which holds the same high level linguistic principles but differ in details. A very simple idea to take advantage of heterogeneous structures is to design a predictor which can predict a more accurate target structure based on the input, the less accurate target structure and complementary structures. This idea is very close to stacked learning (Wolpert, 1992), which is well developed for ensemble learning, and successfully applied to some NLP tasks, e.g. dependency parsing (Nivre and McDonald, 2008; Torres Martins et al., 2008).

Formally speaking, our idea is to include two “levels” of processing. The first level includes one

AS ⇒ u:44;	CD ⇒ m:134;
DEC ⇒ u:83;	DEV ⇒ u:7;
DEG ⇒ u:123;	ETC ⇒ u:9;
LB ⇒ p:1;	NT ⇒ t:98;
OD ⇒ m:41;	PU ⇒ w:552;
SP ⇒ u:1;	VC ⇒ v:32;
VE ⇒ v:13;	BA ⇒ p:2; d:1;
CS ⇒ c:3; d:1;	DT ⇒ r:15; b:1;
MSP ⇒ c:2; u:1;	PN ⇒ r:53; n:2;
CC ⇒ c:73; p:5; v:2;	M ⇒ q:101; n:11; v:1;
LC ⇒ f:51; Ng:3; v:1; u:1;	P ⇒ p:133; v:4; c:2; Vg:1;
VA ⇒ a:57; i:4; z:2; ad:1;	NR ⇒ ns:170; nr:65; j:23;
b:1;	nt:21; nz:7; n:2; s:1;
VV ⇒ v:382; i:5; a:3; Vg:2;	JJ ⇒ a:43; b:13; n:3; vn:3;
vn:2; n:2; p:2; w:1;	d:2; j:2; f:2; t:2; z:1;
AD ⇒ d:149; c:11; ad:6; z:4;	NN ⇒ n:738; vn:135; v:26;
a:3; v:2; n:1; r:1; m:1; f:1;	j:19; Ng:5; an:5; a:3; r:3; s:3;
t:1;	Ag:2; nt:2; f:2; q:2; i:1; t:1;
	nz:1; b:1;

Table 1: Mapping between CTB and PPD POS Tags.

or more base predictors  $f_1, \dots, f_K$  that are independently built on different training data. The second level processing consists of an inference function  $h$  that takes as input  $\langle \mathbf{x}, f_1(\mathbf{x}), \dots, f_K(\mathbf{x}) \rangle^3$  and outputs a final prediction  $h(\mathbf{x}, f_1(\mathbf{x}), \dots, f_K(\mathbf{x}))$ . The only difference between model ensemble and *annotation ensemble* is that the output spaces of model ensemble are the same while the output spaces of annotation ensemble are different. This framework is general and flexible, in the sense that it assumes almost nothing about the individual systems and take them as black boxes.

### 4.2 A Character-based Tagger

With IOB2 representation (Ramshaw and Marcus, 1995), the problem of joint segmentation and tagging can be regarded as a character classification task. Previous work shows that the character-based approach is an effective method for Chinese lexical processing. Both of our feature- and structure-based stacking models employ base character-based taggers to generate multiple segmentation and tagging results. Our base tagger use a discriminative sequential classifier to predict the POS tag with positional information for each character. Each character can be assigned one of two possible boundary tags: “B” for a character that begins a word and “I” for a character that occurs in the middle of a word. We denote

<sup>3</sup> $\mathbf{x}$  is a given Chinese sentence.

a candidate character token  $c_i$  with a fixed window  $c_{i-2}c_{i-1}c_i c_{i+1}c_{i+2}$ . The following features are used for classification:

- Character unigrams:  $c_k$  ( $i - l \leq k \leq i + l$ )
- Character bigrams:  $c_k c_{k+1}$  ( $i - l \leq k < i + l$ )

### 4.3 Feature-based Stacking

Jiang et al. (2009) introduced a feature-based stacking solution for annotation ensemble. In their solution, an auxiliary tagger  $CTag_{ppd}$  is trained on a complementary corpus, i.e. PPD, to assist the target CTB-style tagging. To refine the character-based tagger  $CTag_{ctb}$ , PPD-style character labels are directly incorporated as new features. The stacking model relies on the ability of discriminative learning method to explore informative features, which play central role to boost the tagging performance. To compare their feature-based stacking model and our structure-based model, we implement a similar system  $CTag_{ppd \rightarrow ctb}$ . Apart from character uni/bigram features, the PPD-style character labels are used to derive the following features to enhance our CTB-style tagger:

- Character label unigrams:  $c_k^{ppd}$  ( $i - l^{ppd} \leq k \leq i + l^{ppd}$ )
- Character label bigrams:  $c_k^{ppd} c_{k+1}^{ppd}$  ( $i - l^{ppd} \leq k < i + l^{ppd}$ )

In the above descriptions,  $l$  and  $l^{ppd}$  are the window sizes of features, which can be tuned on development data.

### 4.4 Structure-based Stacking

We propose a novel structured-based stacking model for the task, in which heterogeneous word structures are used not only to generate features but also to derive a sub-word structure. Our work is inspired by the stacked sub-word tagging model introduced in (Sun, 2011). Their work is motivated by the diversity of heterogeneous models, while our work is motivated by the diversity of heterogeneous annotations. The workflow of our new system is shown in Figure 1. In the first phase, one character-based CTB-style tagger ( $CTag_{ctb}$ ) and one character-based PPD-style tagger ( $CTag_{ppd}$ ) are respectively trained to produce heterogeneous

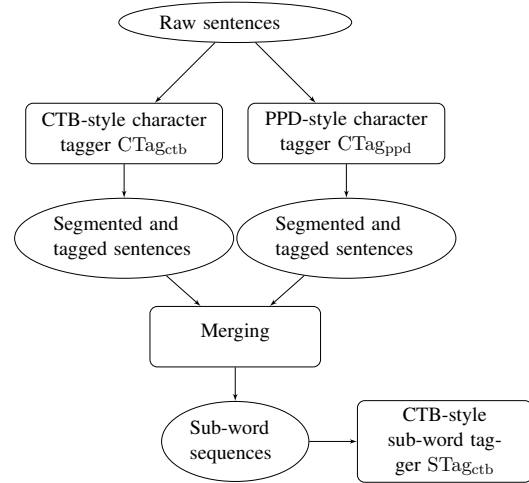


Figure 1: Sub-word tagging based on heterogeneous taggers.

word boundaries. In the second phase, this system first combines the two segmentation and tagging results to get sub-words which maximize the agreement about word boundaries. Finally, a fine-grained sub-word tagger ( $STag_{ctb}$ ) is applied to bracket sub-words into words and also to label their POS tags. We can also apply a PPD-style sub-word tagger. To compare with previous work, we specially concentrate on the *PPD-to-CTB* adaptation.

Following (Sun, 2011), the intermediate sub-word structures is defined to maximize the agreement of  $CTag_{ctb}$  and  $CTag_{ppd}$ . In other words, the goal is to make merged sub-words as large as possible but not overlap with any predicted word produced by the two taggers. If the position between two continuous characters is predicted as a word boundary by any segmenter, this position is taken as a separation position of the sub-word sequence. This strategy makes sure that it is still possible to correctly *re-segment* the strings of which the boundaries are disagreed with by the heterogeneous segmenters in the sub-word tagging stage.

To train the sub-word tagger  $STag_{ctb}$ , features are formed making use of both CTB-style and PPD-style POS tags provided by the character-based taggers. In the following description, “C” refers to the content of a sub-word; “ $T_{ctb}$ ” and “ $T_{ppd}$ ” refers to the positional POS tags generated from  $CTag_{ctb}$  and  $CTag_{ppd}$ ;  $l_C$ ,  $l_T^{ctb}$  and  $l_T^{ppd}$  are the window sizes. For convenience, we denote a sub-word with its con-



text  $\dots s_{i-1} s_i s_{i+1} \dots$ , where  $s_i$  is the current token. The following features are applied:

- Unigram features:  $C(s_k)$  ( $i - l_C \leq k \leq +l_C$ ),  $T_{ctb}(s_k)$  ( $i - l_T^{ctb} \leq k \leq i + l_T^{ctb}$ ),  $T_{ppd}(s_k)$  ( $i - l_T^{ppd} \leq k \leq i + l_T^{ppd}$ )
- Bigram features:  $C(s_k)C(s_{k+1})$  ( $i - l_C \leq k < i + l_C$ ),  $T_{ctb}(s_k)T_{ctb}(s_{k+1})$  ( $i - l_T^{ctb} \leq k < i + l_T^{ctb}$ ),  $T_{ppd}(s_k)T_{ppd}(s_{k+1})$  ( $i - l_T^{ppd} \leq k < i + l_T^{ppd}$ )
- $C(s_{i-1})C(s_{i+1})$  (if  $l_C \geq 1$ ),  $T_{ctb}(s_{i-1})T_{ctb}(s_{i+1})$  (if  $l_T^{ctb} \geq 1$ ),  $T_{ppd}(s_{i-1})T_{ppd}(s_{i+1})$  (if  $l_T^{ppd} \geq 1$ )
- Word formation features: character  $n$ -gram prefixes and suffixes for  $n$  up to 3.

**Cross-validation**  $CTag_{ctb}$  and  $CTag_{ppd}$  are directly trained on the original training data, i.e. the CTB and PPD data. Cross-validation technique has been proved necessary to generate the training data for sub-word tagging, since it deals with the training/test mismatch problem (Sun, 2011). To construct training data for the new heterogeneous sub-word tagger, a 10-fold *cross-validation* on the original CTB data is performed too.

## 5 Data-driven Annotation Conversion

It is possible to acquire high quality labeled data for a specific annotation standard by exploring existing heterogeneous corpora, since the annotations are normally highly compatible. Moreover, the exploitation of additional (pseudo) labeled data aims to reduce the estimation error and enhances a NLP system in a different way from stacking. We therefore expect the improvements are not much overlapping and the combination of them can give a further improvement.

The stacking models can be viewed as annotation converters: They take as input complementary structures and produce as output target structures. In other words, the stacking models actually learn statistical models to transform the lexical representations. We can acquire informative extra samples by processing the PPD data with our stacking models. Though the converted annotations are imperfect, they are still helpful to reduce the estimation error.

**Character-based Conversion** The feature-based stacking model  $CTag_{ppd \rightarrow ctb}$  maps the input character sequence  $\mathbf{c}$  and its PPD-style character label sequence to the corresponding CTB-style character label sequence. This model by itself can be taken as a corpus conversion model to transform a PPD-style analysis to a CTB-style analysis. By processing the auxiliary corpus  $D_{ppd}$  with  $CTag_{ppd \rightarrow ctb}$ , we acquire a new labeled data set  $D'_{ctb} = D_{ppd \rightarrow ctb}^{CTag_{ppd \rightarrow ctb}}$ . We can re-train the  $CTag_{ctb}$  model with both original and converted data  $D_{ctb} \cup D'_{ctb}$ .

**Sub-word-based Conversion** Similarly, the structure-based stacking model can be also taken as a corpus conversion model. By processing the auxiliary corpus  $D_{ppd}$  with  $STag_{ctb}$ , we acquire a new labeled data set  $D''_{ctb} = D_{ppd \rightarrow ctb}^{STag_{ctb}}$ . We can re-train the  $STag_{ctb}$  model with  $D_{ctb} \cup D''_{ctb}$ . If we use the gold PPD-style labels of  $D''_{ctb}$  to extract sub-words, the new model will overfit to the gold PPD-style labels, which are unavailable at test time. To avoid this training/test mismatch problem, we also employ a 10-fold cross validation procedure to add noise.

It is not a new topic to convert corpus from one formalism to another. A well known work is transforming Penn Treebank into resources for various deep linguistic processing, including LTAG (Xia, 1999), CCG (Hockenmaier and Steedman, 2007), HPSG (Miyao et al., 2004) and LFG (Cahill et al., 2002). Such work for corpus conversion mainly leverages rich sets of hand-crafted rules to convert corpora. The construction of linguistic rules is usually time-consuming and the rules are not full coverage. Compared to rule-based conversion, our statistical converters are much easier to build and empirically perform well.

## 6 Experiments

### 6.1 Setting

Previous studies on joint Chinese word segmentation and POS tagging have used the CTB in experiments. We follow this setting in this paper. We use CTB 5.0 as our main corpus and define the training, development and test sets according to (Jiang et al., 2008a; Jiang et al., 2008b; Kruegkrai et al., 2009; Zhang and Clark, 2010; Sun, 2011). Jiang et

al. (2009) present a preliminary study for the annotation adaptation topic, and conduct experiments with the extra PPD data<sup>4</sup>. In other words, the CTB-style annotation is the target analysis while the PPD-style annotation is the complementary/auxiliary analysis. Our experiments for annotation ensemble follows their setting to lead to a fair comparison of our system and theirs. A CRF learning toolkit, wapiti<sup>5</sup> (Lavergne et al., 2010), is used to resolve sequence labeling problems. Among several parameter estimation methods provided by wapiti, our auxiliary experiments indicate that the “rprop-” method works best. Three metrics are used for evaluation: precision (P), recall (R) and balanced f-score (F) defined by  $2PR/(P+R)$ . Precision is the relative amount of correct words in the system output. Recall is the relative amount of correct words compared to the gold standard annotations. A token is considered to be correct if its boundaries match the boundaries of a word in the gold standard and their POS tags are identical.

## 6.2 Results of Stacking

Table 2 summarizes the segmentation and tagging performance of the baseline and different stacking models. The baseline of the character-based joint solver ( $CTag_{ctb}$ ) is competitive, and achieves an f-score of 92.93. By using the character labels from a heterogeneous solver ( $CTag_{ppd}$ ), which is trained on the PPD data set, the performance of this character-based system ( $CTag_{ppd \rightarrow ctb}$ ) is improved to 93.67. This result confirms the importance of a heterogeneous structure. Our structure-based stacking solution is effective and outperforms the feature-based stacking. By better exploiting the heterogeneous word boundary structures, our sub-word tagging model achieves an f-score of 94.03 ( $l_T^{ctb}$  and  $l_T^{ppd}$  are tuned on the development data and both set to 1).

The contribution of the auxiliary tagger is two-fold. On one hand, the heterogeneous solver provides structural information, which is the basis to construct the sub-word sequence. On the other hand, this tagger provides additional POS information, which is helpful for disambiguation. To eval-

<sup>4</sup>[http://ic1.pku.edu.cn/ic1\\_res/](http://ic1.pku.edu.cn/ic1_res/)

<sup>5</sup><http://wapiti.limsi.fr/>

Devel.	P	R	F
$CTag_{ctb}$	93.28%	92.58%	92.93
$CTag_{ppd \rightarrow ctb}$	93.89%	93.46%	93.67
$STag_{ctb}$	94.07%	93.99%	94.03

Table 2: Performance of different stacking models on the development data.

uate these two contributions, we do another experiment by just using the heterogeneous word boundary structures without the POS information. The f-score of this type of sub-word tagging is 93.73. This result indicates that both the word boundary and POS information are helpful.

## 6.3 Learning Curves

We do additional experiments to evaluate the effect of heterogeneous features as the amount of PPD data is varied. Table 3 summarizes the f-score change. The feature-based model works well only when a considerable amount of heterogeneous data is available. When a small set is added, the performance is even lower than the baseline (92.93). The structure-based stacking model is more robust and obtains consistent gains regardless of the size of the complementary data.

#CTB	#PPD	PPD $\rightarrow$ CTB	
		$CTag$	$STag$
18104	7381	92.21	93.26
18104	14545	93.22	93.82
18104	21745	93.58	93.96
18104	28767	93.55	93.87
18104	35996	93.67	94.03
9052	9052	92.10	92.40

Table 3: F-scores relative to sizes of training data. Sizes (shown in column #CTB and #PPD) are numbers of sentences in each training corpus.

## 6.4 Results of Annotation Conversion

The stacking models can be viewed as data-driven annotation converting models. However they are not trained on “real” labeled samples. Although the target representation (CTB-style analysis in our case) is gold standard, the input representation (PPD-style analysis in our case) is labeled by a automatic tagger  $CTag_{ppd}$ . To make clear whether these stacking

models trained with noisy inputs can *tolerate* perfect inputs, we evaluate the two stacking models on our manually converted data. The accuracies presented in Table 4 indicate that though the conversion models are learned by applying noisy data, they can refine target tagging with gold auxiliary tagging. Another interesting thing is that the gold PPD-style analysis does not help the sub-word tagging model as much as the character tagging model.

	Auto PPD	Gold PPD
CTag <sub>ppd→ctb</sub>	93.69	95.19
STag <sub>ctb</sub>	94.14	94.70

Table 4: F-scores with gold PPD-style tagging on the manually converted data.

### 6.5 Results of Re-training

Table 5 shows accuracies of re-trained models. Note that a sub-word tagger is built on character taggers, so when we re-train a sub-word system, we should consider whether or not re-training base character taggers. The error rates decrease as automatically converted data is added to the training pool, especially for the character-based tagger CTag<sub>ctb</sub>. When the base CTB-style tagging is improved, the final tagging is improved in the end. The re-training does not help the sub-word tagging much; the improvement is very modest.

CTag <sub>ctb</sub>	STag <sub>ctb</sub>	P(%)	R(%)	F
$D_{ctb} \cup D'_{ctb}$	--	94.46	94.06	94.26
$D_{ctb} \cup D'_{ctb}$	$D_{ctb}$	94.61	94.43	94.52
$D_{ctb}$	$D_{ctb} \cup D''_{ctb}$	94.05	94.08	94.06
$D_{ctb} \cup D'_{ctb}$	$D_{ctb} \cup D''_{ctb}$	94.71	94.53	94.62

Table 5: Performance of re-trained models on the development data.

### 6.6 Comparison to the State-of-the-Art

Table 6 summarizes the tagging performance of different systems. The baseline of the character-based tagger is competitive, and achieve an f-score of 93.41. By better using the heterogeneous word boundary structures, our sub-word tagging model achieves an f-score of 94.36. Both character and sub-word tagging model can be enhanced with automatically converted corpus. With the pseudo labeled

data, the performance goes up to 94.11 and 94.68. These results are also better than the best published result on the same data set that is reported in (Jiang et al., 2009).

Test	P	R	F
(Sun, 2011)	--	--	94.02
(Jiang et al., 2009)	--	--	94.02
(Wang et al., 2011)	--	--	94.18 <sup>6</sup>
Character model	93.31%	93.51%	93.41
+Re-training	93.93%	94.29%	94.11
Sub-word model	94.10%	94.62%	94.36
+Re-training	<b>94.42%</b>	<b>94.93%</b>	<b>94.68</b>

Table 6: Performance of different systems on the test data.

## 7 Conclusion

Our theoretical and empirical analysis of two representative popular corpora highlights two essential characteristics of heterogeneous annotations which are explored to reduce approximation and estimation errors for Chinese word segmentation and POS tagging. We employ stacking models to incorporate features derived from heterogeneous analysis and apply them to convert heterogeneous labeled data for re-training. The appropriate application of heterogeneous annotations leads to a significant improvement (a relative error reduction of 11%) over the best performance for this task. Although our discussion is for a specific task, the key idea to leverage heterogeneous annotations to reduce the approximation error with stacking models and the estimation error with automatically converted corpora is very general and applicable to other NLP tasks.

## Acknowledgement

This work is mainly finished when the first author was in Saarland University and DFKI. At that time, this author was funded by DFKI and German Academic Exchange Service (DAAD). While working in Peking University, both author are supported by NSFC (61170166) and National High-Tech R&D Program (2012AA011101).

<sup>6</sup>This result is achieved with much unlabeled data, which is different from our setting.

## References

- Aoife Cahill, Mairead Mccarthy, Josef Van Genabith, and Andy Way. 2002. Automatic annotation of the penn treebank with lfg f-structure information. In *Proceedings of the LREC Workshop on Linguistic Knowledge Acquisition and Representation: Bootstrapping Annotated Language Data, Las Palmas, Canary Islands*, pages 8–15.
- Julia Hockenmaier and Mark Steedman. 2007. Ccgbank: A corpus of ccg derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.
- Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lü. 2008a. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL-08: HLT*, pages 897–904, Columbus, Ohio, June. Association for Computational Linguistics.
- Wenbin Jiang, Haitao Mi, and Qun Liu. 2008b. Word lattice reranking for Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 385–392, Manchester, UK, August. Coling 2008 Organizing Committee.
- Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic adaptation of annotation standards: Chinese word segmentation and pos tagging – a case study. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 522–530, Suntec, Singapore, August. Association for Computational Linguistics.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun’ichi Kazama, Yiyou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and pos tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 513–521, Suntec, Singapore, August. Association for Computational Linguistics.
- Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. pages 504–513, July.
- Yusuke Miyao, Takashi Ninomiya, and Jun ichi Tsujii. 2004. Corpus-oriented grammar development for acquiring a head-driven phrase structure grammar from the penn treebank. In *IJCNLP*, pages 684–693.
- Heew Tou Ng and Jin Kiat Low. 2004. Chinese part-of-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 277–284, Barcelona, Spain, July. Association for Computational Linguistics.
- Zheng-Yu Niu, Haifeng Wang, and Hua Wu. 2009. Exploiting heterogeneous treebanks for parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 46–54, Suntec, Singapore, August. Association for Computational Linguistics.
- Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio, June. Association for Computational Linguistics.
- Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In David Yarowsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 82–94, Somerset, New Jersey. Association for Computational Linguistics.
- Weiwei Sun and Jia Xu. 2011. Enhancing Chinese word segmentation using unlabeled data. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 970–979, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Weiwei Sun, Rui Wang, and Yi Zhang. 2010. Discriminative parse reranking for Chinese with homogeneous and heterogeneous annotations. In *Proceedings of Joint Conference on Chinese Language Processing (CIPS-SIGHAN)*, Beijing, China, August.
- Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1211–1219, Beijing, China, August. Coling 2010 Organizing Committee.
- Weiwei Sun. 2011. A stacked sub-word model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1385–1394, Portland, Oregon, USA, June. Association for Computational Linguistics.
- André Filipe Torres Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 157–166, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Yiyou Wang, Jun’ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving chinese word segmentation and pos tagging with semi-supervised methods using large

- auto-analyzed data. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 309–317, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- David H. Wolpert. 1992. Original contribution: Stacked generalization. *Neural Netw.*, 5:241–259, February.
- Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of Natural Language Processing Pacific Rim Symposium*, pages 398–403.
- Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pages 888–896, Columbus, Ohio, June. Association for Computational Linguistics.
- Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 843–852, Cambridge, MA, October. Association for Computational Linguistics.

# Capturing Paradigmatic and Syntagmatic Lexical Relations: Towards Accurate Chinese Part-of-Speech Tagging

Weiwei Sun<sup>†\*</sup> and Hans Uszkoreit<sup>‡</sup>

<sup>†</sup>Institute of Computer Science and Technology, Peking University

<sup>†</sup>Saarbrücken Graduate School of Computer Science

<sup>‡‡</sup>Department of Computational Linguistics, Saarland University

<sup>‡‡</sup>Language Technology Lab, DFKI GmbH

[ws@pku.edu.cn](mailto:ws@pku.edu.cn), [uszkoreit@dfki.de](mailto:uszkoreit@dfki.de)

## Abstract

From the perspective of structural linguistics, we explore paradigmatic and syntagmatic lexical relations for Chinese POS tagging, an important and challenging task for Chinese language processing. Paradigmatic lexical relations are explicitly captured by word clustering on large-scale unlabeled data and are used to design new features to enhance a discriminative tagger. Syntagmatic lexical relations are implicitly captured by constituent parsing and are utilized via system combination. Experiments on the Penn Chinese Treebank demonstrate the importance of both paradigmatic and syntagmatic relations. Our linguistically motivated approaches yield a relative error reduction of 18% in total over a state-of-the-art baseline.

## 1 Introduction

In grammar, a part-of-speech (POS) is a linguistic category of words, which is generally defined by the syntactic or morphological behavior of the word in question. Automatically assigning POS tags to words plays an important role in parsing, word sense disambiguation, as well as many other NLP applications. Many successful tagging algorithms developed for English have been applied to many other languages as well. In some cases, the methods work well without large modifications, such as for German. But a number of augmentations and changes become necessary when dealing with highly inflected or agglutinative languages, as well as analytic languages, of which Chinese is the focus

---

This work is mainly finished when this author (corresponding author) was in Saarland University and DFKI.

of this paper. The Chinese language is characterized by the lack of formal devices such as morphological tense and number that often provide important clues for syntactic processing tasks. While state-of-the-art tagging systems have achieved accuracies above 97% on English, Chinese POS tagging has proven to be more challenging and obtained accuracies about 93-94% (Tseng et al., 2005b; Huang et al., 2007, 2009; Li et al., 2011).

It is generally accepted that Chinese POS tagging often requires more sophisticated language processing techniques that are capable of drawing inferences from more subtle linguistic knowledge. From a linguistic point of view, meaning arises from the differences between linguistic units, including words, phrases and so on, and these differences are of two kinds: paradigmatic (concerning substitution) and syntagmatic (concerning positioning). The distinction is a key one in structuralist semiotic analysis. Both paradigmatic and syntagmatic lexical relations have a great impact on POS tagging, because the *value* of a word is determined by the two relations. Our error analysis of a state-of-the-art Chinese POS tagger shows that the lack of both paradigmatic and syntagmatic lexical knowledge accounts for a large part of tagging errors.

This paper is concerned with capturing paradigmatic and syntagmatic lexical relations to advance the state-of-the-art of Chinese POS tagging. First, we employ unsupervised word clustering to explore paradigmatic relations that are encoded in large-scale unlabeled data. The word clusters are then explicitly utilized to design new features for POS tagging. Second, we study the possible impact of syntagmatic relations on POS tagging by comparatively analyzing a (syntax-free) sequential tagging model

and a (syntax-based) chart parsing model. Inspired by the analysis, we employ a full parser to implicitly capture syntagmatic relations and propose a *Bootstrap Aggregating* (Bagging) model to combine the complementary strengths of a sequential tagger and a parser.

We conduct experiments on the Penn Chinese Treebank and Chinese Gigaword. We implement a discriminative sequential classification model for POS tagging which achieves the state-of-the-art accuracy. Experiments show that this model are significantly improved by word cluster features in accuracy across a wide range of conditions. This confirms the importance of the paradigmatic relations. We then present a comparative study of our tagger and the Berkeley parser, and show that the combination of the two models can significantly improve tagging accuracy. This demonstrates the importance of the syntagmatic relations. Cluster-based features and the Bagging model result in a relative error reduction of 18% in terms of the word classification accuracy.

## 2 State-of-the-Art

### 2.1 Previous Work

Many algorithms have been applied to computationally assigning POS labels to English words, including hand-written rules, generative HMM tagging and discriminative sequence labeling. Such methods have been applied to many other languages as well. In some cases, the methods work well without large modifications, such as German POS tagging. But a number of augmentations and changes became necessary when dealing with Chinese that has little, if any, inflectional morphology. While state-of-the-art tagging systems have achieved accuracies above 97% on English, Chinese POS tagging has proven to be more challenging and obtains accuracies about 93-94% (Tseng et al., 2005b; Huang et al., 2007, 2009; Li et al., 2011).

Both discriminative and generative models have been explored for Chinese POS tagging (Tseng et al., 2005b; Huang et al., 2007, 2009). Tseng et al. (2005a) introduced a maximum entropy based model, which includes morphological features for unknown word recognition. Huang et al. (2007) and Huang et al. (2009) mainly focused on the gener-

ative HMM models. To enhance a HMM model, Huang et al. (2007) proposed a re-ranking procedure to include extra morphological and syntactic features, while Huang et al. (2009) proposed a latent variable inducing model. Their evaluations on the Chinese Treebank show that Chinese POS tagging obtains an accuracy of about 93-94%.

### 2.2 Our Discriminative Sequential Model

According to the *ACL Wiki*, all state-of-the-art English POS taggers are based on discriminative sequence labeling models, including structure perceptron (Collins, 2002; Shen et al., 2007), maximum entropy (Toutanova et al., 2003) and SVM (Gimnez and Mrquez, 2004). A discriminative learner is easy to be extended with arbitrary features and therefore suitable to recognize more new words. Moreover, a majority of the POS tags are locally dependent on each other, so the Markov assumption can well captures the syntactic relations among words. Discriminative learning is also an appropriate solution for Chinese POS tagging, due to its flexibility to include knowledge from multiple linguistic sources.

To deeply analyze the POS tagging problem for Chinese, we implement a discriminative sequential model. A first order linear-chain CRF model is used to resolve the sequential classification problem. We choose the CRF learning toolkit *wapiti*<sup>1</sup> (Lavergne et al., 2010) to train models. In our experiments, we employ a feature set which draws upon information sources such as word forms and characters that constitute words. To conveniently illustrate, we denote a word in focus with a fixed window  $w_{-2}w_{-1}ww_{+1}w_{+2}$ , where  $w$  is the current token. Our features includes:

Word unigrams:  $w_{-2}, w_{-1}, w, w_{+1}, w_{+2}$ ;  
Word bigrams:  $w_{-2}w_{-1}, w_{-1}w, ww_{+1}, w_{+1}w_{+2}$ ;  
In order to better handle unknown words, we extract morphological features: character  $n$ -gram prefixes and suffixes for  $n$  up to 3.

### 2.3 Evaluation

#### 2.3.1 Setting

Penn Chinese Treebank (CTB) (Xue et al., 2005) is a popular data set to evaluate a number of Chinese NLP tasks, including word segmentation (Sun and

<sup>1</sup><http://wapiti.limsi.fr/>

Xu, 2011), POS tagging (Huang et al., 2007, 2009), constituency parsing (Zhang and Clark, 2009; Wang et al., 2006) and dependency parsing (Zhang and Clark, 2008; Huang and Sagae, 2010; Li et al., 2011). In this paper, we use CTB 6.0 as the labeled data for the study. The corpus was collected during different time periods from different sources with a diversity of topics. In order to obtain a representative split of data sets, we define the training, development and test sets following two settings. To compare our tagger with the state-of-the-art, we conduct an experiment using the data setting of (Huang et al., 2009). For detailed analysis and evaluation, we conduct further experiments following the setting of the CoNLL 2009 shared task. The setting is provided by the principal organizer of the CTB project, and considers many annotation details. This setting is more robust for evaluating Chinese language processing algorithms.

### 2.3.2 Overall Performance

Table 1 summarizes the per token classification accuracy (Acc.) of our tagger and results reported in (Huang et al., 2009). Huang et al. (2009) introduced a bigram HMM model with latent variables (*Bigram HMM-LA* in the table) for Chinese tagging. Compared to earlier work (Tseng et al., 2005a; Huang et al., 2007), this model achieves the state-of-the-art accuracy. Despite of simplicity, our discriminative POS tagging model achieves a state-of-the-art performance, even better.

System	Acc.
Trigram HMM (Huang et al., 2009)	93.99%
Bigram HMM-LA (Huang et al., 2009)	94.53%
Our tagger	94.69%

Table 1: Tagging accuracies on the test data (setting 1).

## 2.4 Motivating Analysis

For the following experiments, we only report results on the development data of the CoNLL setting.

### 2.4.1 Correlating Tagging Accuracy with Word Frequency

Table 2 summarizes the prediction accuracy on the development data with respect to the word frequency on the training data. To avoid overestimating the tagging accuracy, these statistics exclude all

punctuations. From this table, we can see that words with low frequency, especially the out-of-vocabulary (OOV) words, are hard to label. However, when a word is very frequently used, its behavior is very complicated and therefore hard to predict. A typical example of such words is the language-specific function word “的.” This analysis suggests that a main topic to enhance Chinese POS tagging is to bridge the gap between the infrequent words and frequent words.

Freq.	Acc.
0	83.55%
1-5	89.31%
6-10	90.20%
11-100	94.88%
101-1000	96.26%
1001-	93.65%

Table 2: Tagging accuracies relative to word frequency.

### 2.4.2 Correlating Tagging Accuracy with Span Length

A word projects its grammatical property to its maximal projection and it syntactically governs all words under the span of its maximal projection. The words under the span of current token thus reflect its syntactic behavior and good clues for POS tagging. Table 3 shows the tagging accuracies relative to the length of the spans. We can see that with the increase of the number of words governed by the token, the difficulty of its POS prediction increase. This analysis suggests that syntagmatic lexical relations plays a significant role in POS tagging, and sometimes words located far from the current token affect its tagging much.

Len.	Acc.
1-2	93.79%
3-4	93.39%
5-6	92.19%
7-	94.18%

Table 3: Tagging accuracies relative to span length.

## 3 Capturing Paradigmatic Relations via Word Clustering

To bridge the gap between high and low frequency words, we employ word clustering to acquire



the knowledge about paradigmatic lexical relations from large-scale texts. Our work is also inspired by the successful application of word clustering to named entity recognition (Miller et al., 2004) and dependency parsing (Koo et al., 2008).

### 3.1 Word Clustering

Word clustering is a technique for partitioning sets of words into subsets of syntactically or semantically similar words. It is a very useful technique to capture paradigmatic or substitutional similarity among words.

#### 3.1.1 Clustering Algorithms

Various clustering techniques have been proposed, some of which, for example, perform automatic word clustering optimizing a maximum-likelihood criterion with iterative clustering algorithms. In this paper, we focus on distributional word clustering that is based on the assumption that words that appear in similar contexts (especially surrounding words) tend to have similar meanings. They have been successfully applied to many NLP problems, such as language modeling.

**Brown Clustering** Our first choice is the bottom-up agglomerative word clustering algorithm of (Brown et al., 1992) which derives a hierarchical clustering of words from unlabeled data. This algorithm generates a hard clustering – each word belongs to exactly one cluster. The input to the algorithm is sequences of words  $w_1, \dots, w_n$ . Initially, the algorithm starts with each word in its own cluster. As long as there are at least two clusters left, the algorithm merges the two clusters that maximizes the quality of the resulting clustering. The quality is defined based on a class-based bigram language model as follows.

$$P(w_i|w_1, \dots, w_{i-1}) \approx p(C(w_i)|C(w_{i-1}))p(w_i|C(w_i))$$

where the function  $C$  maps a word  $w$  to its class  $C(w)$ . We use a publicly available package<sup>2</sup> (Liang et al., 2005) to train this model.

**MKCLS Clustering** We also do experiments by using another popular clustering method based on

<sup>2</sup><http://cs.stanford.edu/~pliang/software/brown-cluster-1.2.zip>

the exchange algorithm (Kneser and Ney, 1993). The objective function is maximizing the likelihood  $\prod_{i=1}^n P(w_i|w_1, \dots, w_{i-1})$  of the training data given a partially class-based bigram model of the form

$$P(w_i|w_1, \dots, w_{i-1}) \approx p(C(w_i)|w_{i-1})p(w_i|C(w_i))$$

We use the publicly available implementation MKCLS<sup>3</sup> (Och, 1999) to train this model.

We choose to work with these two algorithms considering their prior success in other NLP applications. However, we expect that our approach can function with other clustering algorithms.

#### 3.1.2 Data

Chinese Gigaword is a comprehensive archive of newswire text data that has been acquired over several years by the Linguistic Data Consortium (LDC). The large-scale unlabeled data we use in our experiments comes from the Chinese Gigaword (LDC2005T14). We choose the Mandarin news text, i.e. Xinhua newswire. This data covers all news published by Xinhua News Agency (the largest news agency in China) from 1991 to 2004, which contains over 473 million characters.

#### 3.1.3 Pre-processing: Word Segmentation

Different from English and other Western languages, Chinese is written without explicit word delimiters such as space characters. To find the basic language units, i.e. words, segmentation is a necessary pre-processing step for word clustering. Previous research shows that character-based segmentation models trained on labeled data are reasonably accurate (Sun, 2010). Furthermore, as shown in (Sun and Xu, 2011), appropriate string knowledge acquired from large-scale unlabeled data can significantly enhance a supervised model, especially for the prediction of out-of-vocabulary (OOV) words. In this paper, we employ such supervised and semi-supervised segmenters<sup>4</sup> to process raw texts.

### 3.2 Improving Tagging with Cluster Features

Our discriminative sequential tagger is easy to be extended with arbitrary features and therefore suitable to explore additional features derived from other

<sup>3</sup><http://code.google.com/p/giza-pp/>

<sup>4</sup><http://www.coli.uni-saarland.de/~wsun/ccws.tgz>

sources. We propose to use of word clusters as substitutes for word forms to assist the POS tagger. We are relying on the ability of the discriminative learning method to explore informative features, which play a central role in boosting the tagging performance. 5 clustering-based uni/bi-gram features are added:  $w_{-1}$ ,  $w$ ,  $w_{+1}$ ,  $w_{-1}w$ ,  $w_{-1}w_{+1}$ .

### 3.3 Evaluation

Features	Data	Brown	MKCLS
Baseline	CoNLL	94.48%	
+c100	+1991-1995(S)	94.77%	94.83%
+c500	+1991-1995(S)	94.84%	94.93%
+c1000	+1991-1995(S)	--	94.95%
+c100	+1991-1995(SS)	94.90%	94.97%
+c500	+1991-1995(SS)	94.94%	94.88%
+c1000	+1991-1995(SS)	94.89%	94.94%
+c100	+1991-2000(SS)	94.82%	94.93%
+c500	+1991-2000(SS)	94.92%	94.99%
+c1000	+1991-2000(SS)	94.90%	95.00%
+c100	+1991-2004(SS)	--	94.87%
+c500	+1991-2004(SS)	--	<b>95.02%</b>
+c1000	+1991-2004(SS)	--	94.97%

Table 4: Tagging accuracies with different features. *S*: supervised segmentation; *SS*: semi-supervised segmentation.

Table 4 summarizes the tagging results on the development data with different feature configurations. In this table, the symbol “+” in the *Features* column means current configuration contains both the baseline features and new cluster-based features; the number is the total number of the clusters; the symbol “+” in the *Data* column means which portion of the Gigaword data is used to cluster words; the symbol “S” and “SS” in parentheses denote (s)upervised and (s)emi-(s)upervised word segmentation. For example, “+1991-2000(S)” means the data from 1991 to 2000 are processed by a supervised segmenter and used for clustering. From this table, we can clearly see the impact of word clustering features on POS tagging. The new features lead to substantial improvements over the strong supervised baseline. Moreover, these increases are consistent regardless of the clustering algorithms. Both clustering algorithms contributes to the overall performance equivalently. A natural strategy for extending current experiments is to include both clustering results together, or to include more than one cluster granularity. However, we find no further improvement. For

each clustering algorithm, there are not much differences among different sizes of the total clustering numbers. When a comparable amount of unlabeled data (five years’ data) is used, the further increase of the unlabeled data for clustering does not lead to much changes of the tagging performance.

### 3.4 Learning Curves

Size	Baseline	+Cluster
4.5K	90.10%	91.93%
9K	92.91%	93.94%
13.5K	93.88%	94.60%
18K	94.24%	94.77%

Table 5: Tagging accuracies relative to sizes of training data. Size=#sentences in the training corpus.

We do additional experiments to evaluate the effect of the derived features as the amount of labeled training data is varied. We also use the “+c500(MKCLS)+1991-2004(SS)” setting for these experiments. Table 5 summarizes the accuracies of the systems when trained on smaller portions of the labeled data. We can see that the new features obtain consistent gains regardless of the size of the training set. The error is reduced significantly on all data sets. In other words, the word cluster features can significantly reduce the amount of labeled data required by the learning algorithm. The relative reduction is greatest when smaller amounts of the labeled data are used, and the effect lessens as more labeled data is added.

### 3.5 Analysis

Word clustering derives paradigmatic relational information from unlabeled data by grouping words into different sets. As a result, the contribution of word clustering to POS tagging is two-fold. On the one hand, word clustering captures and abstracts context information. This new linguistic *knowledge* is thus helpful to better correlate a word in a certain context to its POS tag. On the other hand, the clustering of the OOV words to some extent fights the sparse data problem by correlating an OOV word with in-vocabulary (IV) words through their classes. To evaluate the two contributions of the word clustering, we limit entries of the clustering lexicon to only contain IV words, i.e. words appearing in the training corpus. Using this constrained lexicon,

we train a new “+c500(MKCLS)+1991-2004(SS)” model and report its prediction power in Table 6. The gap between the baseline and +*IV clustering* models can be viewed as the contribution of the first effect, while the gap between the +*IV clustering* and +*All clustering* models can be viewed as the second contribution. This result indicates that the improved predictive power partially comes from the new interpretation of a POS tag through a clustering, and partially comes from its memory of OOV words that appears in the unlabeled data.

	Baseline	+IV Clustering	+All clustering
Acc.	94.48%	94.70%(↑0.22)	95.02%(↑0.32)

Table 6: Tagging accuracies with IV clustering.

Table 7 shows the recall of OOV words on the development data set. Only the word types appearing more than 10 times are reported. The recall of all OOV words are improved, especially of proper nouns (NR) and common verbs (VV). Another interesting fact is that almost all of them are content words. This table is also helpful to understand the impact of the clustering information on the prediction of OOV words.

## 4 Capturing Syntagmatic Relations via Constituency Parsing

Syntactic analysis, especially the full and deep one, reflects syntagmatic relations of words and phrases of sentences. We present a series of empirical studies of the tagging results of our *syntax-free* sequential tagger and a *syntax-based* chart parser<sup>5</sup>, aiming at illuminating more precisely the impact of information about phrase-structures on POS tagging. The analysis is helpful to understand the role of syntagmatic lexical relations in POS prediction.

### 4.1 Comparing Tagging and PCFG-LA Parsing

The majority of the state-of-the-art constituent parsers are based on generative PCFG learning, with lexicalized (Collins, 2003; Charniak, 2000) or latent annotation (PCFG-LA) (Matsuzaki et al., 2005; Petrov et al., 2006; Petrov and Klein, 2007) refinements. Compared to lexicalized parsers, the PCFG-LA parsers leverages on an automatic procedure to

<sup>5</sup>Both the tagger and the parser are trained on the same portion from CTB.

	#Words	Baseline	+Clustering	$\Delta$
AD	21	33.33%	42.86%	<
CD	249	97.99%	98.39%	<
JJ	86	3.49%	26.74%	<
NN	1028	91.05%	91.34%	<
NR	863	81.69%	88.76%	<
NT	25	60.00%	68.00%	<
VA	15	33.33%	53.33%	<
VV	402	67.66%	72.39%	<

Table 7: The tagging recall of OOV words.

learn refined grammars and are therefore more robust to parse non-English languages that are not well studied. For Chinese, a PCFG-LA parser achieves the state-of-the-art performance and defeat many other types of parsers (Zhang and Clark, 2009). For full parsing, the Berkeley parser<sup>6</sup>, an open source implementation of the PCFG-LA model, is used for experiments. Table 8 shows their overall and detailed performance.

#### 4.1.1 Content Words vs. Function Words

Table 8 gives a detailed comparison regarding different word types. For each type of word, we report the accuracy of both solvers and compare the difference. The majority of the words that are better labeled by the tagger are content words, including nouns (NN, NR, NT), numbers (CD, OD), predicates (VA, VC, VE), adverbs (AD), nominal modifiers (JJ), and so on. In contrast, most of the words that are better predicted by the parser are function words, including most particles (DEC, DEG, DER, DEV, AS, MSP), prepositions (P, BA) and coordinating conjunction (CC).

#### 4.1.2 Open Classes vs. Close Classes

POS can be divided into two broad supercategories: closed class types and open class types. Open classes accept the addition of new morphemes (words), through such processes as compounding, derivation, inflection, coining, and borrowing. On the other hand closed classes are those that have relatively fixed membership. For example, nouns and verbs are open classes because new nouns and verbs are continually coined or borrowed from other languages, while *DEC/DEG* are two closed classes because only the function word “的” is assigned to

<sup>6</sup><http://code.google.com/p/berkeleyparser/>

	Parser<Tagger	Parser>Tagger	
♠ AD	94.15<94.71	♡ AS	98.54>98.44
♠ CD	94.66<97.52	♡ BA	96.15>92.52
	CS 91.12<92.12	♡ CC	93.80>90.58
	ETC 99.65<100.0	♡ DEC	85.78>81.22
♠ JJ	81.35<84.65	♡ DEG	88.94>85.96
	LB 91.30<93.18	♡ DER	80.95>77.42
	LC 96.29<97.08	♡ DEV	84.89>74.78
	M 95.62<96.94	DT	98.28>98.05
♠ NN	93.56<94.95	♡ MSP	91.30>90.14
♠ NR	89.84<95.07	♡ P	96.26>94.56
♠ NT	96.70<97.26	VV	91.99>91.87
♠ OD	81.06<86.36		
	PN 98.10<98.15		
	SB 95.36<96.77		
	SP 61.70<68.89		
♠ VA	81.27<84.25	Overall	
♠ VC	95.91<97.67	Tagger:	94.48%
♠ VE	97.12<98.48	Parser:	93.69%

Table 8: Tagging accuracies of relative to word classes.

them. The discriminative model can conveniently include many features, especially features related to the word formation, which are important to predict words of open classes. Table 9 summarizes the tagging accuracies relative to IV and OOV words. On the whole, the Berkeley parser processes IV words slightly better than our tagger, but processes OOV words significantly worse. The numbers in this table clearly shows the main weakness of the Berkeley parser is the the predictive power of the OOV words.

	IV	OOV
Tagger	95.22%	81.59%
Parser	95.38%	64.77%

Table 9: Tagging accuracies of the IV and OOV words.

#### 4.1.3 Local Disambiguation vs. Global Disambiguation

Closed class words are generally function words that tend to occur frequently and often have structuring uses in grammar. These words have little lexical meaning or have ambiguous meaning, but instead serve to express grammatical relationships with other words within a sentence. They signal the structural relationships that words have to one another and are the glue that holds sentences together. Thus, they serve as important elements to the structures of sentences. The disambiguation of these

words normally require more syntactic clues, which is very hard and inappropriate for a sequential tagger to capture. Based on global grammatical inference of the whole sentence, the full parser is relatively good at dealing with structure related ambiguities.

We conclude that discriminative sequential tagging model can better capture local syntactic and morphological information, while the full parser can better capture global syntactic structural information. The discriminative tagging model are limited by the Markov assumption and inadequate to correctly label structure related words.

#### 4.2 Enhancing POS Tagging via Bagging

The diversity analysis suggests that we may improve parsing by simply combining the tagger and the parser. Bootstrap aggregating (Bagging) is a machine learning ensemble meta-algorithm to improve classification and regression models in terms of stability and classification accuracy (Breiman, 1996). It also reduces variance and helps to avoid overfitting. We introduce a Bagging model to integrate different POS tagging models. In the training phase, given a training set  $D$  of size  $n$ , our model generates  $m$  new training sets  $D_i$  of size  $63.2\% \times n$  by sampling examples from  $D$  without replacement. Namely no example will be repeated in each  $D_i$ . Each  $D_i$  is separately used to train a tagger and a parser. Using this strategy, we can get  $2m$  weak solvers. In the tagging phase, the  $2m$  models outputs  $2m$  tagging results, each word is assigned one POS label. The final tagging is the voting result of these  $2m$  labels. There may be equal number of different tags. In this case, our system prefer the first label they met.

#### 4.3 Evaluation

We evaluate our combination model on the same data set used above. Figure 1 shows the influence of  $m$  in the Bagging algorithm. Because each new data set  $D_i$  in bagging algorithm is generated by a random procedure, the performance of all Bagging experiments are not the same. To give a more stable evaluation, we repeat 5 experiments for each  $m$  and show the averaged accuracy. We can see that the Bagging model taking both sequential tagging and chart parsing models as basic systems outperform the baseline systems and the Bagging model taking either model in isolation as basic systems. An

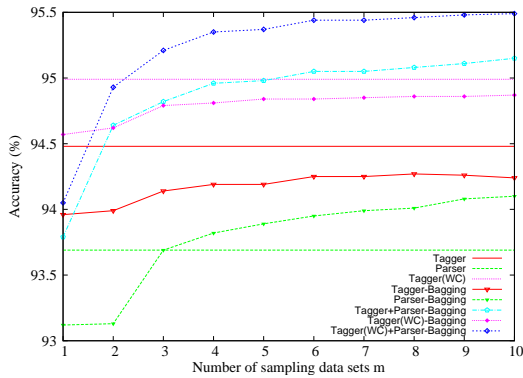


Figure 1: Tagging accuracies of Bagging models. *Tagger-Bagging* and *Tagger(WC)-Bagging* means that the Bagging system built on the tagger with and without word clusters. *Parser-Bagging* is named in the same way. *Tagger+Paser-Bagging* and *Tagger(WC)+Paser-Bagging* means that the Bagging systems are built on both tagger and parser.

interesting phenomenon is that the Bagging method can also improve the parsing model, but there is a decrease while only combining taggers.

## 5 Combining Both

We have introduced two separate improvements for Chinese POS tagging, which capture different types of lexical relations. We therefore expect further improvement by combining both enhancements, since their contributions to the task is different. We still use a Bagging model to integrate the discriminative tagger and the Berkeley parser. The only difference between current experiment and previous experiment is that the sub-tagging models are trained with help of word clustering features. Figure 1 also shows the performance of the new Bagging model on the development data set. We can see that the improvements that come from two ways, namely capturing syntagmatic and paradigmatic relations, are not much overlapping and the combination of them gives more.

Table 10 shows the performance of different systems evaluated on the test data. The final result is remarkable. The word clustering features and the Bagging model result in a relative error reduction of 18% in terms of the classification accuracy. The significant improvement of the POS tagging also help successive language processing. Results in Table

Systems	Acc.
Baseline	94.33%
Tagger(WC)	94.85%
Tagger+Parser( $m = 15$ )	94.96%
Tagger(WC)+Parser( $m = 15$ )	95.34%

Table 10: Tagging accuracies on the test data (CoNLL).

11 indicate that the parsing accuracy of the Berkeley parser can be simply improved by inputting the Berkeley parser with the POS Bagging results. Although the combination with a syntax-based tagger is very effective, there are two weaknesses: (1) a syntax-based model relies on linguistically rich syntactic annotations that are not easy to acquire; (2) a syntax-based model is computationally expensive which causes efficiency difficulties.

Tagger	LP	LR	F
Berkeley	82.71%	80.57%	81.63
Bagging( $m = 15$ )	82.96%	81.44%	82.19

Table 11: Parsing accuracies on the test data. (CoNLL)

## 6 Conclusion

We hold a view of structuralist linguistics and study the impact of paradigmatic and syntagmatic lexical relations on Chinese POS tagging. First, we harvest word partition information from large-scale raw texts to capture paradigmatic relations and use such knowledge to enhance a supervised tagger via feature engineering. Second, we comparatively analyze syntax-free and syntax-based models and employ a Bagging model to integrate a sequential tagger and a chart parser to capture syntagmatic relations that have a great impact on non-local disambiguation. Both enhancements significantly improve the state-of-the-art of Chinese POS tagging. The final model results in an error reduction of 18% over a state-of-the-art baseline.

## Acknowledgement

This work is mainly finished when the first author was in Saarland University and DFKI. At that time, this author was funded by DFKI and German Academic Exchange Service (DAAD). While working in Peking University, the first author is supported by NSFC (61170166) and National High-Tech R&D Program (2012AA011101).



## References

- Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.
- Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479. URL <http://portal.acm.org/citation.cfm?id=176313.176316>.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8. Association for Computational Linguistics. URL <http://www.aclweb.org/anthology/W02-1001>.
- Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.
- Jesús Giménez and Lluís Màrquez. 2004. Svmtool: A general pos tagger generator based on support vector machines. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 43–46.
- Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1077–1086. Association for Computational Linguistics, Uppsala, Sweden. URL <http://www.aclweb.org/anthology/P10-1110>.
- Zhongqiang Huang, Vladimir Eidelman, and Mary Harper. 2009. Improving a simple bigram hmm part-of-speech tagger by latent annotation and self-training. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 213–216. Association for Computational Linguistics, Boulder, Colorado. URL <http://www.aclweb.org/anthology/N/N09/N09-2054>.
- Zhongqiang Huang, Mary Harper, and Wen Wang. 2007. Mandarin part-of-speech tagging and discriminative reranking. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 1093–1102. Association for Computational Linguistics, Prague, Czech Republic. URL <http://www.aclweb.org/anthology/D/D07/D07-1117>.
- Reinhard Kneser and Hermann Ney. 1993. Improved clustering techniques for class-based statistical language modeling. In *Proceedings of the European Conference on Speech Communication and Technology (Eurospeech)*.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603. Association for Computational Linguistics, Columbus, Ohio. URL <http://www.aclweb.org/anthology/P/P08/P08-1068>.
- Thomas Lavergne, Olivier Cappé, and François Yvon. 2010. Practical very large scale CRFs. pages 504–513. URL <http://www.aclweb.org/anthology/P10-1052>.
- Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for Chinese pos tagging and dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1180–1191. Association for Computational Linguistics, Edinburgh, Scotland, UK. URL <http://www.aclweb.org/anthology/D11-1109>.
- Percy Liang, Michael Collins, and Percy Liang. 2005. Semi-supervised learning for natural language. In *Master's thesis, MIT*.
- Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic cfg with latent annotations. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05*, pages 75–82. Association for Computational Linguistics, Stroudsburg,

- PA, USA. URL <http://dx.doi.org/10.3115/1219840.1219850>.
- Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 337–342. Association for Computational Linguistics, Boston, Massachusetts, USA.
- Franz Josef Och. 1999. An efficient method for determining bilingual word classes. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, EACL '99, pages 71–76. Association for Computational Linguistics, Stroudsburg, PA, USA. URL <http://dx.doi.org/10.3115/977035.977046>.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics, Sydney, Australia.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 404–411. Association for Computational Linguistics, Rochester, New York.
- Libin Shen, Giorgio Satta, and Aravind Joshi. 2007. Guided learning for bidirectional sequence classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 760–767. Association for Computational Linguistics, Prague, Czech Republic. URL <http://www.aclweb.org/anthology/P07-1096>.
- Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1211–1219. Coling 2010 Organizing Committee, Beijing, China. URL <http://www.aclweb.org/anthology/C10-2139>.
- Weiwei Sun and Jia Xu. 2011. Enhancing Chinese word segmentation using unlabeled data. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 970–979. Association for Computational Linguistics, Edinburgh, Scotland, UK. URL <http://www.aclweb.org/anthology/D11-1090>.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1, NAACL '03*, pages 173–180. Association for Computational Linguistics, Stroudsburg, PA, USA. URL <http://dx.doi.org/10.3115/1073445.1073478>.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005a. A conditional random field word segmenter. In *In Fourth SIGHAN Workshop on Chinese Language Processing*.
- Huihsin Tseng, Daniel Jurafsky, and Christopher Manning. 2005b. Morphological features help pos tagging of unknown words across language varieties. In *The Fourth SIGHAN Workshop on Chinese Language Processing*.
- Mengqiu Wang, Kenji Sagae, and Teruko Mitamura. 2006. A fast, accurate deterministic parser for Chinese. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 425–432. Association for Computational Linguistics, Sydney, Australia. URL <http://www.aclweb.org/anthology/P06-1054>.
- Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The penn Chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2):207–238.
- Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based

and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571. Association for Computational Linguistics, Honolulu, Hawaii. URL <http://www.aclweb.org/anthology/D08-1059>.

Yue Zhang and Stephen Clark. 2009. Transition-based parsing of the Chinese treebank using a global discriminative model. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 162–171. Association for Computational Linguistics, Paris, France. URL <http://www.aclweb.org/anthology/W09-3825>.



# Fast Online Training with Frequency-Adaptive Learning Rates for Chinese Word Segmentation and New Word Detection

Xu Sun<sup>†</sup>, Houfeng Wang<sup>‡</sup>, Wenjie Li<sup>†</sup>

<sup>†</sup>Department of Computing, The Hong Kong Polytechnic University

<sup>‡</sup>Key Laboratory of Computational Linguistics (Peking University), Ministry of Education, China  
{csxsun, cswjli}@comp.polyu.edu.hk    wanghf@pku.edu.cn

## Abstract

We present a joint model for Chinese word segmentation and new word detection. We present high dimensional new features, including word-based features and enriched edge (label-transition) features, for the joint modeling. As we know, training a word segmentation system on large-scale datasets is already costly. In our case, adding high dimensional new features will further slow down the training speed. To solve this problem, we propose a new training method, adaptive online gradient descent based on feature frequency information, for very fast online training of the parameters, even given large-scale datasets with high dimensional features. Compared with existing training methods, our training method is an order magnitude faster in terms of training time, and can achieve equal or even higher accuracies. The proposed fast training method is a general purpose optimization method, and it is not limited in the specific task discussed in this paper.

## 1 Introduction

Since Chinese sentences are written as continuous sequences of characters, segmenting a character sequence into words is normally the first step in the pipeline of Chinese text processing. The major problem of Chinese word segmentation is the ambiguity. Chinese character sequences are normally ambiguous, and new words (out-of-vocabulary words) are a major source of the ambiguity. A typical category of new words is named entities, including organization names, person names, location names, and so on.

In this paper, we present high dimensional new features, including word-based features and enriched edge (label-transition) features, for the joint modeling of Chinese word segmentation (CWS) and new word detection (NWD). While most of the state-of-the-art CWS systems used semi-Markov conditional random fields or latent variable conditional random fields, we simply use a single first-order conditional random fields (CRFs) for the joint modeling. The semi-Markov CRFs and latent variable CRFs relax the Markov assumption of CRFs to express more complicated dependencies, and therefore to achieve higher disambiguation power. Alternatively, our plan is not to relax Markov assumption of CRFs, but to exploit more complicated dependencies via using refined high-dimensional features. The advantage of our choice is the simplicity of our model. As a result, our CWS model can be more efficient compared with the heavier systems, and with similar or even higher accuracy because of using refined features.

As we know, training a word segmentation system on large-scale datasets is already costly. In our case, adding high dimensional new features will further slow down the training speed. To solve this challenging problem, we propose a new training method, adaptive online gradient descent based on feature frequency information (ADF), for very fast word segmentation with new word detection, even given large-scale datasets with high dimensional features. In the proposed training method, we try to use more refined learning rates. Instead of using a single learning rate (a scalar) for all weights, we extend the learning rate scalar to a learning rate vector based on feature frequency information in the updating. By doing so, each weight has

its own learning rate adapted on feature frequency information. We will show that this can significantly improve the convergence speed of online learning. We approximate the learning rate vector based on *feature frequency information in the updating process*. Our proposal is based on the intuition that a feature with higher frequency in the training process should be with a learning rate that is decayed faster. Based on this intuition, we will show the formalized training algorithm later. We will show in experiments that our solution is an order magnitude faster compared with exiting learning methods, and can achieve equal or even higher accuracies.

The contribution of this work is as follows:

- We propose a general purpose fast online training method, ADF. The proposed training method requires only a few passes to complete the training.
- We propose a joint model for Chinese word segmentation and new word detection.
- Compared with prior work, our system achieves better accuracies on both word segmentation and new word detection.

## 2 Related Work

First, we review related work on word segmentation and new word detection. Then, we review popular online training methods, in particular stochastic gradient descent (SGD).

### 2.1 Word Segmentation and New Word Detection

Conventional approaches to Chinese word segmentation treat the problem as a sequential labeling task (Xue, 2003; Peng et al., 2004; Tseng et al., 2005; Asahara et al., 2005; Zhao et al., 2010). To achieve high accuracy, most of the state-of-the-art systems are heavy probabilistic systems using semi-Markov assumptions or latent variables (Andrew, 2006; Sun et al., 2009b). For example, one of the state-of-the-art CWS system is the latent variable conditional random field (Sun et al., 2008; Sun and Tsujii, 2009) system presented in Sun et al. (2009b). It is a heavy probabilistic model and it is slow in training. A few other state-of-the-art CWS systems are using semi-Markov perceptron methods or voting systems based on multiple semi-Markov

perceptron segmenters (Zhang and Clark, 2007; Sun, 2010). Those semi-Markov perceptron systems are moderately faster than the heavy probabilistic systems using semi-Markov conditional random fields or latent variable conditional random fields. However, a disadvantage of the perceptron style systems is that they can not provide probabilistic information.

On the other hand, new word detection is also one of the important problems in Chinese information processing. Many statistical approaches have been proposed (J. Nie and Jin, 1995; Chen and Bai, 1998; Wu and Jiang, 2000; Peng et al., 2004; Chen and Ma, 2002; Zhou, 2005; Goh et al., 2003; Fu and Luke, 2004; Wu et al., 2011). New word detection is normally considered as a separate process from segmentation. There were studies trying to solve this problem jointly with CWS. However, the current studies are limited. Integrating the two tasks would benefit both segmentation and new word detection. Our method provides a convenient framework for doing this. Our new word detection is not a stand-alone process, but an integral part of segmentation.

### 2.2 Online Training

The most representative online training method is the SGD method. The SGD uses a small randomly-selected subset of the training samples to approximate the gradient of an objective function. The number of training samples used for this approximation is called the batch size. By using a smaller batch size, one can update the parameters more frequently and speed up the convergence. The extreme case is a batch size of 1, and it gives the maximum frequency of updates, which we adopt in this work. Then, the model parameters are updated in such a way:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \gamma_t \nabla_{\mathbf{w}_t} \mathcal{L}_{stoch}(\mathbf{z}_i, \mathbf{w}_t), \quad (1)$$

where  $t$  is the update counter,  $\gamma_t$  is the learning rate, and  $\mathcal{L}_{stoch}(\mathbf{z}_i, \mathbf{w}_t)$  is the stochastic loss function based on a training sample  $\mathbf{z}_i$ .

There were accelerated versions of SGD, including stochastic meta descent (Vishwanathan et al., 2006) and periodic step-size adaptation online learning (Hsu et al., 2009). Compared with those two methods, our proposal is fundamentally

different. Those two methods are using 2nd-order gradient (Hessian) information for accelerated training, while our accelerated training method does not need such 2nd-order gradient information, which is costly and complicated. Our ADF training method is based on feature frequency adaptation, and there is no prior work on using feature frequency information for accelerating online training.

Other online training methods includes averaged SGD with feedback (Sun et al., 2010; Sun et al., 2011), latent variable perceptron training (Sun et al., 2009a), and so on. Those methods are less related to this paper.

### 3 System Architecture

#### 3.1 A Joint Model Based on CRFs

First, we briefly review CRFs. CRFs are proposed as a method for structured classification by solving “the label bias problem” (Lafferty et al., 2001). Assuming a feature function that maps a pair of observation sequence  $\mathbf{x}$  and label sequence  $\mathbf{y}$  to a global feature vector  $\mathbf{f}$ , the probability of a label sequence  $\mathbf{y}$  conditioned on the observation sequence  $\mathbf{x}$  is modeled as follows (Lafferty et al., 2001):

$$P(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{\exp\{\mathbf{w}^\top \mathbf{f}(\mathbf{y}, \mathbf{x})\}}{\sum_{\mathbf{y}'} \exp\{\mathbf{w}^\top \mathbf{f}(\mathbf{y}', \mathbf{x})\}}, \quad (2)$$

where  $\mathbf{w}$  is a parameter vector.

Given a training set consisting of  $n$  labeled sequences,  $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)$ , for  $i = 1 \dots n$ , parameter estimation is performed by maximizing the objective function,

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^n \log P(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w}) - R(\mathbf{w}). \quad (3)$$

The first term of this equation represents a conditional log-likelihood of a training data. The second term is a regularizer for reducing overfitting. We employed an L2 prior,  $R(\mathbf{w}) = \frac{\|\mathbf{w}\|^2}{2\sigma^2}$ . In what follows, we denote the conditional log-likelihood of each sample  $\log P(\mathbf{y}_i|\mathbf{x}_i, \mathbf{w})$  as  $\ell(\mathbf{z}_i, \mathbf{w})$ . The final objective function is as follows:

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^n \ell(\mathbf{z}_i, \mathbf{w}) - \frac{\|\mathbf{w}\|^2}{2\sigma^2}. \quad (4)$$

Since no word list can be complete, new word identification is an important task in Chinese NLP. New words in input text are often incorrectly segmented into single-character or other very short words (Chen and Bai, 1998). This phenomenon will also undermine the performance of Chinese word segmentation. We consider here new word detection as an integral part of segmentation, aiming to improve both segmentation and new word detection: detected new words are added to the word list lexicon in order to improve segmentation. Based on our CRF word segmentation system, we can compute a probability for each segment. When we find some word segments are of reliable probabilities yet they are not in the existing word list, we then treat those “confident” word segments as new words and add them into the existing word list. Based on preliminary experiments, we treat a word segment as a new word if its probability is larger than 0.5. Newly detected words are re-incorporated into word segmentation for improving segmentation accuracies.

#### 3.2 New Features

Here, we will describe high dimensional new features for the system.

##### 3.2.1 Word-based Features

There are two ideas in deriving the refined features. The first idea is to exploit word features for node features of CRFs. Note that, although our model is a Markov CRF model, we can still use word features to learn word information in the training data. To derive word features, first of all, our system automatically collect a list of word unigrams and bigrams from the training data. To avoid overfitting, we only collect the word unigrams and bigrams whose frequency is larger than 2 in the training set. This list of word unigrams and bigrams are then used as a unigram-dictionary and a bigram-dictionary to generate word-based *unigram* and *bigram* features. The word-based features are indicator functions that fire when the local character sequence matches a word unigram or bigram occurred in the training data. The word-based feature templates derived for the label  $y_i$  are as follows:

- $\text{unigram1}(\mathbf{x}, y_i) \leftarrow [x_{j,i}, y_i]$ , if the character sequence  $x_{j,i}$  matches a word  $w \in \mathbb{U}$ ,

with the constraint  $i - 6 < j < i$ . The item  $x_{j,i}$  represents the character sequence  $x_j \dots x_i$ .  $\mathbb{U}$  represents the unigram-dictionary collected from the training data.

- $\text{unigram2}(\mathbf{x}, y_i) \leftarrow [x_{i,k}, y_i]$ , if the character sequence  $x_{i,k}$  matches a word  $w \in \mathbb{U}$ , with the constraint  $i < k < i + 6$ .
- $\text{bigram1}(\mathbf{x}, y_i) \leftarrow [x_{j,i-1}, x_{i,k}, y_i]$ , if the word bigram candidate  $[x_{j,i-1}, x_{i,k}]$  hits a word bigram  $[w_i, w_j] \in \mathbb{B}$ , and satisfies the aforementioned constraints on  $j$  and  $k$ .  $\mathbb{B}$  represents the word bigram dictionary collected from the training data.
- $\text{bigram2}(\mathbf{x}, y_i) \leftarrow [x_{j,i}, x_{i+1,k}, y_i]$ , if the word bigram candidate  $[x_{j,i}, x_{i+1,k}]$  hits a word bigram  $[w_i, w_j] \in \mathbb{B}$ , and satisfies the aforementioned constraints on  $j$  and  $k$ .

We also employ the traditional character-based features. For each label  $y_i$ , we use the feature templates as follows:

- Character unigrams locating at positions  $i - 2$ ,  $i - 1$ ,  $i$ ,  $i + 1$  and  $i + 2$
- Character bigrams locating at positions  $i - 2$ ,  $i - 1$ ,  $i$  and  $i + 1$
- Whether  $x_j$  and  $x_{j+1}$  are identical, for  $j = i - 2, \dots, i + 1$
- Whether  $x_j$  and  $x_{j+2}$  are identical, for  $j = i - 3, \dots, i + 1$

The latter two feature templates are designed to detect character or word reduplication, a morphological phenomenon that can influence word segmentation in Chinese.

### 3.2.2 High Dimensional Edge Features

The node features discussed above are based on a single label  $y_i$ . CRFs also have edge features that are based on label transitions. The second idea is to incorporate local observation information of  $\mathbf{x}$  in edge features. For traditional implementation of CRF systems (e.g., the HCRF package), usually the edges features contain only the information of  $y_{i-1}$  and  $y_i$ , and without the information of

the observation sequence (i.e.,  $\mathbf{x}$ ). The major reason for this simple realization of edge features in traditional CRF implementation is for reducing the dimension of features. Otherwise, there can be an explosion of edge features in some tasks. For example, in part-of-speech tagging tasks, there can be more than 40 labels and more than 1,600 types of label transitions. Therefore, incorporating local observation information into the edge feature will result in an explosion of edge features, which is 1,600 times larger than the number of feature templates.

Fortunately, for our task, the label set is quite small,  $\mathbb{Y} = \{\text{B}, \text{I}, \text{E}\}$ <sup>1</sup>. There are only nine possible label transitions:  $\mathbb{T} = \mathbb{Y} \times \mathbb{Y}$  and  $|\mathbb{T}| = 9$ .<sup>2</sup> As a result, the feature dimension will have nine times increase over the feature templates, if we incorporate local observation information of  $\mathbf{x}$  into the edge features. In this way, we can effectively combine observation information of  $\mathbf{x}$  with label transitions  $y_{i-1}y_i$ . We simply used the same templates of node features for deriving the new edge features. We found adding new edge features significantly improves the disambiguation power of our model.

## 4 Adaptive Online Gradient Descent based on Feature Frequency Information

As we will show in experiments, the training of the CRF model with high-dimensional new features is quite expensive, and the existing training method is not good enough. To solve this issue, we propose a fast online training method: adaptive online gradient descent based on feature frequency information (ADF). The proposed method is easy to implement.

For high convergence speed of online learning, we try to use more refined learning rates than the SGD training. Instead of using a single learning rate (a scalar) for all weights, we extend the learning rate scalar to a learning rate vector, which has the same dimension of the weight vector  $\mathbf{w}$ . The learning rate vector is automatically adapted based on feature frequency information. By doing so, each weight

<sup>1</sup>B means *beginning of a word*, I means *inside a word*, and E means *end of a word*. The B, I, E labels have been widely used in previous work of Chinese word segmentation (Sun et al., 2009b).

<sup>2</sup>The operator  $\times$  means a Cartesian product between two sets.

---

**ADF learning algorithm**

---

```

1: procedure ADF( $q, c, \alpha, \beta$ )
2:    $\mathbf{w} \leftarrow 0, t \leftarrow 0, \mathbf{v} \leftarrow 0, \boldsymbol{\gamma} \leftarrow c$ 
3:   repeat until convergence
4:     . Draw a sample  $\mathbf{z}_i$  at random
5:     .  $\mathbf{v} \leftarrow \text{UPDATE}(\mathbf{v}, \mathbf{z}_i)$ 
6:     . if  $t > 0$  and  $t \bmod q = 0$ 
7:       . .  $\boldsymbol{\gamma} \leftarrow \text{UPDATE}(\boldsymbol{\gamma}, \mathbf{v})$ 
8:       . .  $\mathbf{v} \leftarrow 0$ 
9:       .  $\mathbf{g} \leftarrow \nabla_{\mathbf{w}} \mathcal{L}_{stoch}(\mathbf{z}_i, \mathbf{w})$ 
10:      .  $\mathbf{w} \leftarrow \mathbf{w} + \boldsymbol{\gamma} \cdot \mathbf{g}$ 
11:      .  $t \leftarrow t + 1$ 
12:   return  $\mathbf{w}$ 
13:
14: procedure UPDATE( $\mathbf{v}, \mathbf{z}_i$ )
15:   for  $k \in$  features used in sample  $\mathbf{z}_i$ 
16:     .  $\mathbf{v}_k \leftarrow \mathbf{v}_k + 1$ 
17:   return  $\mathbf{v}$ 
18:
19: procedure UPDATE( $\boldsymbol{\gamma}, \mathbf{v}$ )
20:   for  $k \in$  all features
21:     .  $u \leftarrow \mathbf{v}_k / q$ 
22:     .  $\eta \leftarrow \alpha - u(\alpha - \beta)$ 
23:     .  $\boldsymbol{\gamma}_k \leftarrow \eta \boldsymbol{\gamma}_k$ 
24:   return  $\boldsymbol{\gamma}$ 

```

---

Figure 1: The proposed ADF online learning algorithm.  $q, c, \alpha,$  and  $\beta$  are hyper-parameters.  $q$  is an integer representing window size.  $c$  is for initializing the learning rates.  $\alpha$  and  $\beta$  are the upper and lower bounds of a scalar, with  $0 < \beta < \alpha < 1$ .

has its own learning rate, and we will show that this can significantly improve the convergence speed of online learning.

In our proposed online learning method, the update formula is as follows:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \boldsymbol{\gamma}_t \cdot \mathbf{g}_t. \quad (5)$$

The update term  $\mathbf{g}_t$  is the gradient term of a randomly sampled instance:

$$\mathbf{g}_t = \nabla_{\mathbf{w}_t} \mathcal{L}_{stoch}(\mathbf{z}_i, \mathbf{w}_t) = \nabla_{\mathbf{w}_t} \left\{ \ell(\mathbf{z}_i, \mathbf{w}_t) - \frac{\|\mathbf{w}_t\|^2}{2n\sigma^2} \right\}.$$

In addition,  $\boldsymbol{\gamma}_t \in \mathbb{R}_+^f$  is a positive vector-valued learning rate and  $\cdot$  denotes component-wise (Hadamard) product of two vectors.

We learn the learning rate vector  $\boldsymbol{\gamma}_t$  based on *feature frequency information in the updating*

*process*. Our proposal is based on the intuition that a feature with higher frequency in the training process should be with a learning rate that decays faster. In other words, we assume a high frequency feature observed in the training process should have a small learning rate, and a low frequency feature should have a relatively larger learning rate in the training. Our assumption is based on the intuition that a weight with higher frequency is more adequately trained, hence smaller learning rate is preferable for fast convergence.

Given a window size  $q$  (number of samples in a window), we use a vector  $\mathbf{v}$  to record the feature frequency. The  $k$ 'th entry  $\mathbf{v}_k$  corresponds to the frequency of the feature  $k$  in this window. Given a feature  $k$ , we use  $u$  to record the normalized frequency:

$$u = \mathbf{v}_k / q.$$

For each feature, an adaptation factor  $\eta$  is calculated based on the normalized frequency information, as follows:

$$\eta = \alpha - u(\alpha - \beta),$$

where  $\alpha$  and  $\beta$  are the upper and lower bounds of a scalar, with  $0 < \beta < \alpha < 1$ . As we can see, a feature with higher frequency corresponds to a smaller scalar via linear approximation. Finally, the learning rate is updated as follows:

$$\boldsymbol{\gamma}_k \leftarrow \eta \boldsymbol{\gamma}_k.$$

With this setting, different features will correspond to different adaptation factors based on feature frequency information. Our ADF algorithm is summarized in Figure 1.

The ADF training method is efficient, because the additional computation (compared with SGD) is only the derivation of the learning rates, which is simple and efficient. As we know, the regularization of SGD can perform efficiently via the optimization based on sparse features (Shalev-Shwartz et al., 2007). Similarly, the derivation of  $\boldsymbol{\gamma}_t$  can also perform efficiently via the optimization based on sparse features.

#### 4.1 Convergence Analysis

Prior work on convergence analysis of existing online learning algorithms (Murata, 1998; Hsu et

Data	Method	Passes	Train-Time (sec)	NWD Rec	Pre	Rec	CWS F-score
MSR	Baseline	50	4.7e3	72.6	96.3	95.9	96.1
	+ New features	50	1.2e4	75.3	97.2	97.0	97.1
	+ New word detection	50	1.2e4	78.2	97.5	96.9	97.2
	+ ADF training	10	2.3e3	77.5	97.6	97.2	<b>97.4</b>
CU	Baseline	50	2.9e3	68.5	94.0	93.9	93.9
	+ New features	50	7.5e3	68.0	94.4	94.5	94.4
	+ New word detection	50	7.5e3	68.8	94.8	94.5	94.7
	+ ADF training	10	1.5e3	68.8	94.8	94.7	<b>94.8</b>
PKU	Baseline	50	2.2e3	77.2	95.0	94.0	94.5
	+ New features	50	5.2e3	78.4	95.5	94.9	95.2
	+ New word detection	50	5.2e3	79.1	95.8	94.9	95.3
	+ ADF training	10	1.2e3	78.4	95.8	94.9	<b>95.4</b>

Table 2: Incremental evaluations, by incrementally adding *new features* (word features and high dimensional edge features), *new word detection*, and *ADF training* (replacing SGD training with ADF training). Number of passes is decided by empirical convergence of the training methods.

	#W.T.	#Word	#C.T.	#Char
MSR	$8.8 \times 10^4$	$2.4 \times 10^6$	$5 \times 10^3$	$4.1 \times 10^6$
CU	$6.9 \times 10^4$	$1.5 \times 10^6$	$5 \times 10^3$	$2.4 \times 10^6$
PKU	$5.5 \times 10^4$	$1.1 \times 10^6$	$5 \times 10^3$	$1.8 \times 10^6$

Table 1: Details of the datasets. *W.T.* represents *word types*; *C.T.* represents *character types*.

al., 2009) can be extended to the proposed ADF training method. We can show that the proposed ADF learning algorithm has reasonable convergence properties.

When we have the smallest learning rate  $\gamma_{t+1} = \beta\gamma_t$ , the expectation of the obtained  $\mathbf{w}_t$  is

$$E(\mathbf{w}_t) = \mathbf{w}^* + \prod_{m=1}^t (\mathbf{I} - \gamma_0 \beta^m \mathbf{H}(\mathbf{w}^*)) (\mathbf{w}_0 - \mathbf{w}^*),$$

where  $\mathbf{w}^*$  is the optimal weight vector, and  $\mathbf{H}$  is the Hessian matrix of the objective function. The rate of convergence is governed by the largest eigenvalue of the function  $\mathbf{C}_t = \prod_{m=1}^t (\mathbf{I} - \gamma_0 \beta^m \mathbf{H}(\mathbf{w}^*))$ . Then, we can derive a bound of rate of convergence.

**Theorem 1** Assume  $\phi$  is the largest eigenvalue of the function  $\mathbf{C}_t = \prod_{m=1}^t (\mathbf{I} - \gamma_0 \beta^m \mathbf{H}(\mathbf{w}^*))$ . For the proposed ADF training, its convergence rate is bounded by  $\phi$ , and we have

$$\phi \leq \exp \left\{ \frac{\gamma_0 \lambda \beta}{\beta - 1} \right\},$$

where  $\lambda$  is the minimum eigenvalue of  $\mathbf{H}(\mathbf{w}^*)$ .

## 5 Experiments

### 5.1 Data and Metrics

We used benchmark datasets provided by the second International Chinese Word Segmentation Bakeoff to test our proposals. The datasets are from Microsoft Research Asia (MSR), City University of Hongkong (CU), and Peking University (PKU). Details of the corpora are listed in Table 1. We did not use any extra resources such as common surnames, parts-of-speech, and semantics.

Four metrics were used to evaluate segmentation results: recall ( $R$ , the percentage of gold standard output words that are correctly segmented by the decoder), precision ( $P$ , the percentage of words in the decoder output that are segmented correctly), balanced F-score defined by  $2PR/(P + R)$ , and recall of new word detection (NWD recall). For more detailed information on the corpora, refer to Emerson (2005).

### 5.2 Features, Training, and Tuning

We employed the feature templates defined in Section 3.2. The feature sets are huge. There are  $2.4 \times 10^7$  features for the MSR data,  $4.1 \times 10^7$  features for the CU data, and  $4.7 \times 10^7$  features for the PKU data. To generate word-based features, we extracted high-frequency word-based unigram and bigram lists from the training data.

As for training, we performed gradient descent

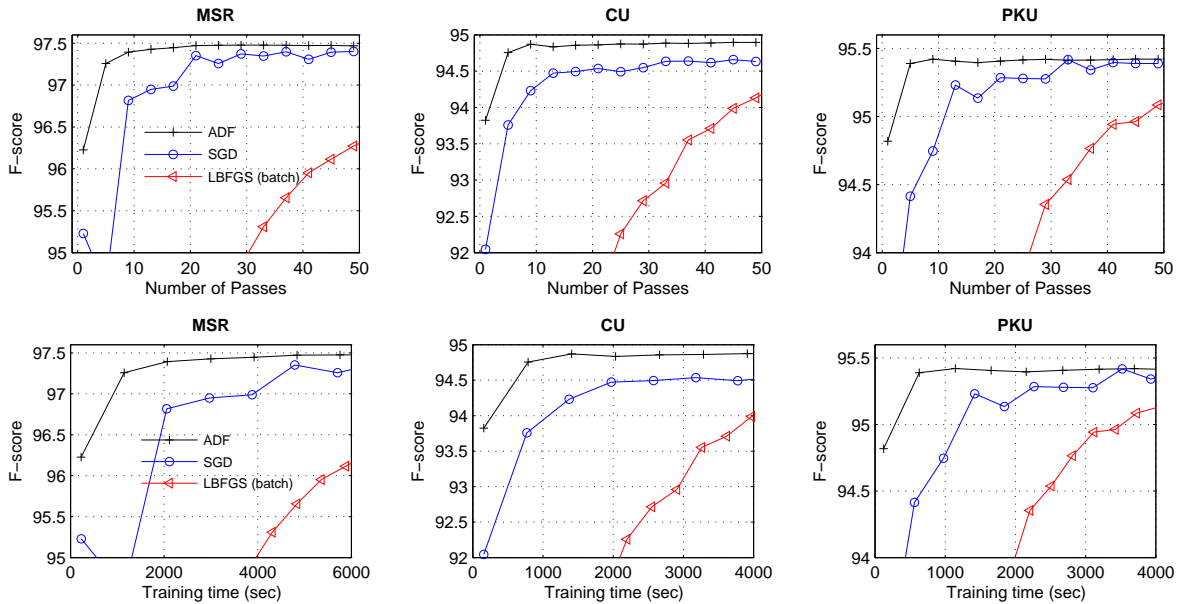


Figure 2: F-score curves on the MSR, CU, and PKU datasets: ADF learning vs. SGD and LBFGS training methods.

with our proposed training method. To compare with existing methods, we chose two popular training methods, a batch training one and an online training one. The batch training method is the Limited-Memory BFGS (LBFGS) method (Nocedal and Wright, 1999). The online baseline training method is the SGD method, which we have introduced in Section 2.2.

For the ADF training method, we need to tune the hyper-parameters  $q$ ,  $c$ ,  $\alpha$ , and  $\beta$ . Based on automatic tuning within the training data (validation in the training data), we found it is proper to set  $q = n/10$  ( $n$  is the number of training samples),  $c = 0.1$ ,  $\alpha = 0.995$ , and  $\beta = 0.6$ . To reduce overfitting, we employed an  $L_2$  Gaussian weight prior (Chen and Rosenfeld, 1999) for all training methods. We varied the  $\sigma$  with different values (e.g., 1.0, 2.0, and 5.0), and finally set the value to 1.0 for all training methods.

### 5.3 Results and Discussion

First, we performed incremental evaluation in this order: Baseline (word segmentation model with SGD training); Baseline + New features; Baseline + New features + New word detection; Baseline + New features + New word detection + ADF training (replacing SGD training). The results are shown in Table 2.

As we can see, the new features improved performance on both word segmentation and new word detection. However, we also noticed that the training cost became more expensive via adding high dimensional new features. Adding new word detection function further improved the segmentation quality and the new word recognition recall. Finally, by using the ADF training method, the training speed is much faster than the SGD training method. The ADF method can achieve empirical optimum in only a few passes, yet with better segmentation accuracies than the SGD training with 50 passes.

To get more details of the proposed training method, we compared it with SGD and LBFGS training methods based on an identical platform, by varying the number of passes. The comparison was based on the same platform: Baseline + New features + New word detection. The F-score curves of the training methods are shown in Figure 2. Impressively, the ADF training method reached empirical convergence in only a few passes, while the SGD and LBFGS training converged much slower, requiring more than 50 passes. The ADF training is about an order magnitude faster than the SGD online training and more than an order magnitude faster than the LBFGS batch training.

Finally, we compared our method with the state-

Data	Method	Prob.	Pre	Rec	F-score
MSR	Best05 (Tseng et al., 2005)	✓	96.2	96.6	96.4
	CRF + rule-system (Zhang et al., 2006)	✓	97.2	96.9	97.1
	Semi-Markov perceptron (Zhang and Clark, 2007)	×	N/A	N/A	97.2
	Semi-Markov CRF (Gao et al., 2007)	✓	N/A	N/A	97.2
	Latent-variable CRF (Sun et al., 2009b)	✓	97.3	97.3	97.3
	<b>Our method (A Single CRF)</b>	✓	97.6	97.2	<b>97.4</b>
CU	Best05 (Tseng et al., 2005)	✓	94.1	94.6	94.3
	CRF + rule-system (Zhang et al., 2006)	✓	95.2	94.9	95.1
	Semi-perceptron (Zhang and Clark, 2007)	×	N/A	N/A	95.1
	Latent-variable CRF (Sun et al., 2009b)	✓	94.7	94.4	94.6
	<b>Our method (A Single CRF)</b>	✓	94.8	94.7	94.8
	PKU	Best05 (Chen et al., 2005)	N/A	95.3	94.6
CRF + rule-system (Zhang et al., 2006)		✓	94.7	95.5	95.1
semi-perceptron (Zhang and Clark, 2007)		×	N/A	N/A	94.5
Latent-variable CRF (Sun et al., 2009b)		✓	95.6	94.8	95.2
<b>Our method (A Single CRF)</b>		✓	95.8	94.9	<b>95.4</b>

Table 3: Comparing our method with the state-of-the-art CWS systems.

of-the-art systems reported in the previous papers. The statistics are listed in Table 3. *Best05* represents the best system of the Second International Chinese Word Segmentation Bakeoff on the corresponding data; *CRF + rule-system* represents confidence-based combination of CRF and rule-based models, presented in Zhang et al. (2006). *Prob.* indicates whether or not the system can provide probabilistic information. As we can see, our method achieved similar or even higher F-scores, compared with the best systems reported in previous papers. Note that, our system is a single Markov model, while most of the state-of-the-art systems are complicated heavy systems, with model-combinations (e.g., voting of multiple segmenters), semi-Markov relaxations, or latent-variables.

## 6 Conclusions and Future Work

In this paper, we presented a joint model for Chinese word segmentation and new word detection. We presented new features, including word-based features and enriched edge features, for the joint modeling. We showed that the new features can improve the performance on the two tasks.

On the other hand, the training of the model, especially with high-dimensional new features, became quite expensive. To solve this problem,

we proposed a new training method, ADF training, for very fast training of CRFs, even given large-scale datasets with high dimensional features. We performed experiments and showed that our new training method is an order magnitude faster than existing optimization methods. Our final system can learn highly accurate models with only a few passes in training. The proposed fast learning method is a general algorithm that is not limited in this specific task. As future work, we plan to apply this fast learning method on other large-scale natural language processing tasks.

## Acknowledgments

We thank Yaozhong Zhang and Weiwei Sun for helpful discussions on word segmentation techniques. The work described in this paper was supported by a Hong Kong RGC Project (No. PolyU 5230/08E), National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), and National Natural Science Foundation of China (No.91024009, No.60973053).

## References

Galen Andrew. 2006. A hybrid markov/semi-markov conditional random field for sequence segmentation.



- In *Proceedings of EMNLP'06*, pages 465–472.
- Masayuki Asahara, Kenta Fukuoka, Ai Azuma, Chooi-Ling Goh, Yotaro Watanabe, Yuji Matsumoto, and Takahashi Tsuzuki. 2005. Combination of machine learning methods for optimum chinese word segmentation. In *Proceedings of The Fourth SIGHAN Workshop*, pages 134–137.
- K.J. Chen and M.H. Bai. 1998. Unknown word detection for chinese by a corpus-based learning method. *Computational Linguistics and Chinese Language Processing*, 3(1):27–44.
- Keh-Jiann Chen and Wei-Yun Ma. 2002. Unknown word extraction for chinese documents. In *Proceedings of COLING'02*.
- Stanley F. Chen and Ronald Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. *Technical Report CMU-CS-99-108, CMU*.
- Aitao Chen, Yiping Zhou, Anne Zhang, and Gordon Sun. 2005. Unigram language model for chinese word segmentation. In *Proceedings of the fourth SIGHAN workshop*, pages 138–141.
- Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *Proceedings of the fourth SIGHAN workshop*, pages 123–133.
- Guohong Fu and Kang-Kwong Luke. 2004. Chinese unknown word identification using class-based lm. In *Proceedings of IJCNLP'04*, volume 3248 of *Lecture Notes in Computer Science*, pages 704–713. Springer.
- Jianfeng Gao, Galen Andrew, Mark Johnson, and Kristina Toutanova. 2007. A comparative study of parameter estimation methods for statistical natural language processing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL'07)*, pages 824–831.
- Chooi-Ling Goh, Masayuki Asahara, and Yuji Matsumoto. 2003. Chinese unknown word identification using character-based tagging and chunking. In Kotaro Funakoshi, Sandra Kbler, and Jahna Otterbacher, editors, *Proceedings of ACL (Companion)'03*, pages 197–200.
- Chun-Nan Hsu, Han-Shen Huang, Yu-Ming Chang, and Yuh-Jye Lee. 2009. Periodic step-size adaptation in second-order gradient descent for single-pass on-line structured learning. *Machine Learning*, 77(2-3):195–224.
- M. Hannan J. Nie and W. Jin. 1995. Unknown word detection and segmentation of chinese using statistical and heuristic knowledge. *Communications of the Chinese and Oriental Languages Information Processing Society*, 5:47C57.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML'01)*, pages 282–289.
- Noboru Murata. 1998. A statistical study of on-line learning. In *On-line learning in neural networks, Cambridge University Press*, pages 63–92.
- Jorge Nocedal and Stephen J. Wright. 1999. Numerical optimization. *Springer*.
- Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *Proceedings of Coling 2004*, pages 562–568, Geneva, Switzerland, Aug 23–Aug 27. COLING.
- Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. 2007. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of ICML'07*.
- Xu Sun and Jun'ichi Tsujii. 2009. Sequential labeling with latent variables: An exact inference algorithm and its efficient approximation. In *Proceedings of EACL'09*, pages 772–780, Athens, Greece, March.
- Xu Sun, Louis-Philippe Morency, Daisuke Okanohara, and Jun'ichi Tsujii. 2008. Modeling latent-dynamic in shallow parsing: A latent conditional model with improved inference. In *Proceedings of COLING'08*, pages 841–848, Manchester, UK.
- Xu Sun, Takuya Matsuzaki, Daisuke Okanohara, and Jun'ichi Tsujii. 2009a. Latent variable perceptron algorithm for structured classification. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pages 1236–1242.
- Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka, and Jun'ichi Tsujii. 2009b. A discriminative latent variable chinese segmenter with hybrid word/character information. In *Proceedings of NAACL-HLT'09*, pages 56–64, Boulder, Colorado, June.
- Xu Sun, Hisashi Kashima, Takuya Matsuzaki, and Naonori Ueda. 2010. Averaged stochastic gradient descent with feedback: An accurate, robust, and fast training method. In *Proceedings of the 10th International Conference on Data Mining (ICDM'10)*, pages 1067–1072.
- Xu Sun, Hisashi Kashima, Ryota Tomioka, and Naonori Ueda. 2011. Large scale real-life action recognition using conditional random fields with stochastic training. In *Proceedings of the 15th Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'11)*.
- Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING'10 (Posters)*, pages 1211–1219. Chinese Information Processing Society of China.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A

- conditional random field word segmenter for sighthan bakeoff 2005. In *Proceedings of The Fourth SIGHAN Workshop*, pages 168–171.
- S.V.N. Vishwanathan, Nicol N. Schraudolph, Mark W. Schmidt, and Kevin P. Murphy. 2006. Accelerated training of conditional random fields with stochastic meta-descent. In *Proceedings of ICML'06*, pages 969–976.
- A. Wu and Z. Jiang. 2000. Statistically-enhanced new word identification in a rule-based chinese system. In *Proceedings of the Second Chinese Language Processing Workshop*, page 46C51, Hong Kong, China.
- Yi-Lun Wu, Chaio-Wen Hsieh, Wei-Hsuan Lin, Chun-Yi Liu, and Liang-Chih Yu. 2011. Unknown word extraction from multilingual code-switching sentences (in chinese). In *Proceedings of ROCLING (Posters)'11*, pages 349–360.
- Nianwen Xue. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, 8(1):29–48.
- Yue Zhang and Stephen Clark. 2007. Chinese segmentation with a word-based perceptron algorithm. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 840–847, Prague, Czech Republic, June. Association for Computational Linguistics.
- Ruiqiang Zhang, Genichiro Kikui, and Eiichiro Sumita. 2006. Subword-based tagging by conditional random fields for chinese word segmentation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 193–196, New York City, USA, June. Association for Computational Linguistics.
- Hai Zhao, Changning Huang, Mu Li, and Bao-Liang Lu. 2010. A unified character-based tagging framework for chinese word segmentation. *ACM Trans. Asian Lang. Inf. Process.*, 9(2).
- Guodong Zhou. 2005. A chunking strategy towards unknown word detection in chinese word segmentation. In Robert Dale, Kam-Fai Wong, Jian Su, and Oi Yee Kwong, editors, *Proceedings of IJCNLP'05*, volume 3651 of *Lecture Notes in Computer Science*, pages 530–541. Springer.

# Verb Classification using Distributional Similarity in Syntactic and Semantic Structures

**Danilo Croce**

University of Tor Vergata  
00133 Roma, Italy  
croce@info.uniroma2.it

**Alessandro Moschitti**

University of Trento  
38123 Povo (TN), Italy  
moschitti@disi.unitn.it

**Roberto Basili**

University of Tor Vergata  
00133 Roma, Italy  
basili@info.uniroma2.it

**Martha Palmer**

University of Colorado at Boulder  
Boulder, CO 80302, USA  
mpalmer@colorado.edu

## Abstract

In this paper, we propose innovative representations for automatic classification of verbs according to mainstream linguistic theories, namely VerbNet and FrameNet. First, syntactic and semantic structures capturing essential lexical and syntactic properties of verbs are defined. Then, we design advanced similarity functions between such structures, i.e., semantic tree kernel functions, for exploiting distributional and grammatical information in Support Vector Machines. The extensive empirical analysis on VerbNet class and frame detection shows that our models capture meaningful syntactic/semantic structures, which allows for improving the state-of-the-art.

## 1 Introduction

Verb classification is a fundamental topic of computational linguistics research given its importance for understanding the role of verbs in conveying semantics of natural language (NL). Additionally, generalization based on verb classification is central to many NL applications, ranging from shallow semantic parsing to semantic search or information extraction. Currently, a lot of interest has been paid to two verb categorization schemes: VerbNet (Schuler, 2005) and FrameNet (Baker et al., 1998), which has also fostered production of many automatic approaches to predicate argument extraction.

Such work has shown that syntax is necessary for helping to predict the roles of verb arguments and consequently their verb sense (Gildea and Jurafsky, 2002; Pradhan et al., 2005; Gildea and Palmer, 2002). However, the definition of models for optimally combining lexical and syntactic constraints is

still far from being accomplished. In particular, the exhaustive design and experimentation of lexical and syntactic features for learning verb classification appears to be computationally problematic. For example, the verb **order** can belong to the two VerbNet classes:

- The class 60.1, i.e., *order someone to do something* as shown in: *The Illinois Supreme Court **ordered** the commission to audit Commonwealth Edison's construction expenses and refund any unreasonable expenses.*
- The class 13.5.1: *order or request something like* in: *... Michelle blabs about it to a sandwich man while **ordering** lunch over the phone.*

Clearly, the syntactic realization can be used to discern the cases above but it would not be enough to correctly classify the following verb occurrence: *.. ordered the lunch to be delivered ..* in Verb class 13.5.1. For such a case, selectional restrictions are needed. These have also been shown to be useful for semantic role classification (Zapirain et al., 2010). Note that their coding in learning algorithms is rather complex: we need to take into account syntactic structures, which may require an exponential number of syntactic features (i.e., all their possible substructures). Moreover, these have to be enriched with lexical information to trigger lexical preference.

In this paper, we tackle the problem above by studying innovative representations for automatic verb classification according to VerbNet and FrameNet. We define syntactic and semantic structures capturing essential lexical and syntactic properties of verbs. Then, we apply similarity between

such structures, i.e., kernel functions, which can also exploit distributional lexical semantics, to train automatic classifiers. The basic idea of such functions is to compute the similarity between two verbs in terms of all the possible substructures of their syntactic frames. We define and automatically extract a lexicalized approximation of the latter. Then, we apply kernel functions that jointly model structural and lexical similarity so that syntactic properties are combined with generalized lexemes. The nice property of kernel functions is that they can be used in place of the scalar product of feature vectors to train algorithms such as Support Vector Machines (SVMs). This way SVMs can learn the association between syntactic (sub-) structures whose lexical arguments are generalized and target verb classes, i.e., they can also learn selectional restrictions.

We carried out extensive experiments on verb class and frame detection which showed that our models greatly improve on the state-of-the-art (up to about 13% of relative error reduction). Such results are nicely assessed by manually inspecting the most important substructures used by the classifiers as they largely correlate with syntactic frames defined in VerbNet.

In the rest of the paper, Sec. 2 reports on related work, Sec. 3 and Sec. 4 describe previous and our models for syntactic and semantic similarity, respectively, Sec. 5 illustrates our experiments, Sec. 6 discusses the output of the models in terms of error analysis and important structures and finally Sec. 7 derives the conclusions.

## 2 Related work

Our target task is verb classification but at the same time our models exploit distributional models as well as structural kernels. The next three subsections report related work in such areas.

**Verb Classification.** The introductory verb classification example has intuitively shown the complexity of defining a comprehensive feature representation. Hereafter, we report on analysis carried out in previous work.

It has been often observed that verb senses tend to show different selectional constraints in a specific argument position and the above verb *order* is a clear example. In the direct object position of the example sentence for the first sense 60.1 of *order*, we found

*commission* in the role PATIENT of the predicate. It clearly satisfies the +ANIMATE/+ORGANIZATION restriction on the PATIENT role. This is not true for the direct object dependency of the alternative sense 13.5.1, which usually expresses the THEME role, with unrestricted type selection. When properly generalized, the direct object information has thus been shown highly predictive about verb sense distinctions.

In (Brown et al., 2011), the so called *dynamic dependency neighborhoods* (DDN), i.e., the set of verbs that are typically collocated with a direct object, are shown to be more helpful than lexical information (e.g., WordNet). The set of typical verbs taking a noun  $n$  as a direct object is in fact a strong characterization for semantic similarity, as all the nouns  $m$  similar to  $n$  tend to collocate with the same verbs. This is true also for other syntactic dependencies, among which the direct object dependency is possibly the strongest cue (as shown for example in (Dligach and Palmer, 2008)).

In order to generalize the above DDN feature, distributional models are ideal, as they are designed to model all the collocations of a given noun, according to large scale corpus analysis. Their ability to capture lexical similarity is well established in WSD tasks (e.g. (Schutze, 1998)), thesauri harvesting (Lin, 1998), semantic role labeling (Croce et al., 2010)) as well as information retrieval (e.g. (Furnas et al., 1988)).

**Distributional Models (DMs).** These models follow the distributional hypothesis (Firth, 1957) and characterize lexical meanings in terms of *context of use*, (Wittgenstein, 1953). By inducing geometrical notions of vectors and norms through corpus analysis, they provide a topological definition of semantic similarity, i.e., distance in a space. DMs can capture the similarity between words such as *delegation*, *deputation* or *company* and *commission*. In case of sense 60.1 of the verb *order*, DMs can be used to suggest that the role PATIENT can be inherited by all these words, as suitable *Organisations*.

In supervised language learning, when few examples are available, DMs support cost-effective lexical generalizations, often outperforming knowledge based resources (such as WordNet, as in (Pantel et al., 2007)). Obviously, the choice of the context

type determines the type of targeted semantic properties. Wider contexts (e.g., entire documents) are shown to suggest topical relations. Smaller contexts tend to capture more specific semantic aspects, e.g. the syntactic behavior, and better capture paradigmatic relations, such as synonymy. In particular, word space models, as described in (Sahlgren, 2006), define contexts as the words appearing in a  $n$ -sized window, centered around a target word. Co-occurrence counts are thus collected in a words-by-words matrix, where each element records the number of times two words co-occur within a single window of word tokens. Moreover, robust weighting schemas are used to smooth counts against too frequent co-occurrence pairs: Pointwise Mutual Information (PMI) scores (Turney and Pantel, 2010) are commonly adopted.

**Structural Kernels.** Tree and sequence kernels have been successfully used in many NLP applications, e.g., parse reranking and adaptation, (Collins and Duffy, 2002; Shen et al., 2003; Toutanova et al., 2004; Kudo et al., 2005; Titov and Henderson, 2006), chunking and dependency parsing, e.g., (Kudo and Matsumoto, 2003; Daumé III and Marcu, 2004), named entity recognition, (Cumby and Roth, 2003), text categorization, e.g., (Cancedda et al., 2003; Gliozzo et al., 2005), and relation extraction, e.g., (Zelenko et al., 2002; Bunescu and Mooney, 2005; Zhang et al., 2006).

Recently, DMs have been also proposed in integrated syntactic-semantic structures that feed advanced learning functions, such as the semantic tree kernels discussed in (Bloehdorn and Moschitti, 2007a; Bloehdorn and Moschitti, 2007b; Mehdad et al., 2010; Croce et al., 2011).

### 3 Structural Similarity Functions

In this paper we model verb classifiers by exploiting previous technology for kernel methods. In particular, we design new models for verb classification by adopting algorithms for structural similarity, known as Smoothed Partial Tree Kernels (SPTKs) (Croce et al., 2011). We define new innovative structures and similarity functions based on LSA.

The main idea of SPTK is rather simple: (i) measuring the similarity between two trees in terms of the number of shared subtrees; and (ii) such number also includes similar fragments whose lexical nodes

are just related (so they can be different). The contribution of (ii) is proportional to the lexical similarity of the tree lexical nodes, where the latter can be evaluated according to distributional models or also lexical resources, e.g., WordNet.

In the following, we define our models based on previous work on LSA and SPTKs.

#### 3.1 LSA as lexical similarity model

Robust representations can be obtained through intelligent dimensionality reduction methods. In LSA the original word-by-context matrix  $M$  is decomposed through Singular Value Decomposition (SVD) (Landauer and Dumais, 1997; Golub and Kahan, 1965) into the product of three new matrices:  $U$ ,  $S$ , and  $V$  so that  $S$  is diagonal and  $M = USV^T$ .  $M$  is then approximated by  $M_k = U_k S_k V_k^T$ , where only the first  $k$  columns of  $U$  and  $V$  are used, corresponding to the first  $k$  greatest singular values. This approximation supplies a way to project a generic term  $w_i$  into the  $k$ -dimensional space using  $W = U_k S_k^{1/2}$ , where each row corresponds to the representation vectors  $\vec{w}_i$ . The original statistical information about  $M$  is captured by the new  $k$ -dimensional space, which preserves the global structure while removing low-variance dimensions, i.e., distribution noise. Given two words  $w_1$  and  $w_2$ , the term similarity function  $\sigma$  is estimated as the cosine similarity between the corresponding projections  $\vec{w}_1, \vec{w}_2$  in the LSA space, i.e.  $\sigma(w_1, w_2) = \frac{\vec{w}_1 \cdot \vec{w}_2}{\|\vec{w}_1\| \|\vec{w}_2\|}$ . This is known as *Latent Semantic Kernel (LSK)*, proposed in (Cristianini et al., 2001), as it defines a positive semi-definite Gram matrix  $G = \sigma(w_1, w_2) \forall w_1, w_2$  (Shawe-Taylor and Cristianini, 2004).  $\sigma$  is thus a valid kernel and can be combined with other kernels, as discussed in the next session.

#### 3.2 Tree Kernels driven by Semantic Similarity

To our knowledge, two main types of tree kernels exploit lexical similarity: the syntactic semantic tree kernel defined in (Bloehdorn and Moschitti, 2007a) applied to constituency trees and the smoothed partial tree kernels (SPTKs) defined in (Croce et al., 2011), which generalizes the former. We report the definition of the latter as we modified it for our purposes. SPTK computes the number of common substructures between two trees  $T_1$  and  $T_2$  without explicitly considering the whole fragment space. Its

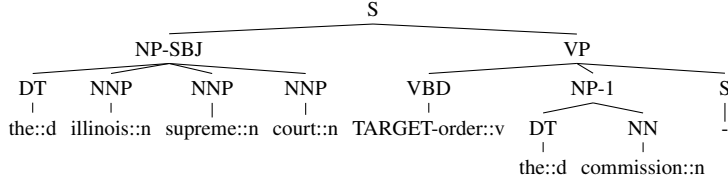


Figure 1: Constituency Tree (CT) representation of verbs.

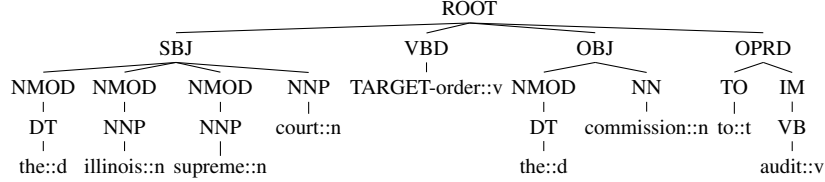


Figure 2: Representation of verbs according to the Grammatical Relation Centered Tree (GRCT)

general equations are reported hereafter:

$$TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2), \quad (1)$$

where  $N_{T_1}$  and  $N_{T_2}$  are the sets of the  $T_1$ 's and  $T_2$ 's nodes, respectively and  $\Delta(n_1, n_2)$  is equal to the number of common fragments rooted in the  $n_1$  and  $n_2$  nodes<sup>1</sup>. The  $\Delta$  function determines the richness of the kernel space and thus induces different tree kernels, for example, the syntactic tree kernel (STK) (Collins and Duffy, 2002) or the partial tree kernel (PTK) (Moschitti, 2006).

The algorithm for SPTK's  $\Delta$  is the following: if  $n_1$  and  $n_2$  are leaves then  $\Delta_\sigma(n_1, n_2) = \mu\lambda\sigma(n_1, n_2)$ ; else

$$\Delta_\sigma(n_1, n_2) = \mu\sigma(n_1, n_2) \times \left( \lambda^2 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1)=l(\vec{I}_2)} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta_\sigma(c_{n_1}(\vec{I}_{1j}), c_{n_2}(\vec{I}_{2j})) \right), \quad (2)$$

where (1)  $\sigma$  is any similarity between nodes, e.g., between their lexical labels; (2)  $\lambda, \mu \in [0, 1]$  are decay factors; (3)  $c_{n_1}(h)$  is the  $h^{th}$  child of the node  $n_1$ ; (4)  $\vec{I}_1$  and  $\vec{I}_2$  are two sequences of indexes, i.e.,  $\vec{I} = (i_1, i_2, \dots, l(I))$ , with  $1 \leq i_1 < i_2 < \dots < i_{l(I)}$ ; and (5)  $d(\vec{I}_1) = \vec{I}_{1l(\vec{I}_1)} - \vec{I}_{11} + 1$  and  $d(\vec{I}_2) = \vec{I}_{2l(\vec{I}_2)} - \vec{I}_{21} + 1$ . Note that, as shown in (Croce et al., 2011), the average running time of SPTK is sub-quadratic in the number of the tree nodes. In the next section we show how we exploit the class of SPTKs, for verb classification.

<sup>1</sup>To have a similarity score between 0 and 1, a normalization in the kernel space, i.e.  $\frac{TK(T_1, T_2)}{\sqrt{TK(T_1, T_1) \times TK(T_2, T_2)}}$  is applied.

## 4 Verb Classification Models

The design of SPTK-based algorithms for our verb classification requires the modeling of two different aspects: (i) a tree representation for the verbs; and (ii) the lexical similarity suitable for the task. We also modified SPTK to apply different similarity functions to different nodes to introduce flexibility.

### 4.1 Verb Structural Representation

The implicit feature space generated by structural kernels and the corresponding notion of similarity between verbs obviously depends on the input structures. In the cases of STK, PTK and SPTK different tree representations lead to engineering more or less expressive linguistic feature spaces.

With the aim of capturing syntactic features, we started from two different parsing paradigms: phrase and dependency structures. For example, for representing the first example of the introduction, we can use the constituency tree (CT) in Figure 1, where the target verb node is enriched with the TARGET label. Here, we apply tree pruning to reduce the computational complexity of tree kernels as it is proportional to the number of nodes in the input trees. Accordingly, we only keep the subtree dominated by the target VP by pruning from it all the S-nodes along with their subtrees (i.e., all nested sentences are removed). To further improve generalization, we lemmatize lexical nodes and add generalized POS-Tags, i.e., noun (n::), verb (v::), adjective (::a), determiner (::d) and so on, to them. This is useful for constraining similarity to be only contributed by lexical pairs of the same grammatical category.

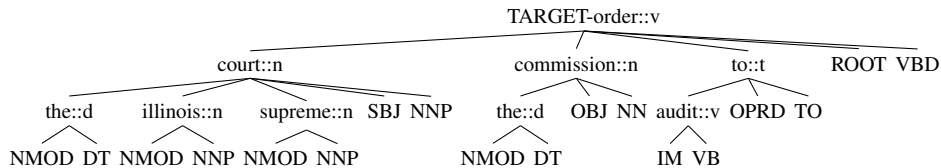


Figure 3: Representation of verbs according to the Lexical Centered Tree (LCT)

To encode dependency structure information in a tree (so that we can use it in tree kernels), we use (i) lexemes as nodes of our tree, (ii) their dependencies as edges between the nodes and (iii) the dependency labels, e.g., grammatical functions (GR), and POS-Tags, again as tree nodes. We designed two different tree types: (i) in the first type, GR are central nodes from which dependencies are drawn and all the other features of the central node, i.e., lexical surface form and its POS-Tag, are added as additional children. An example of the GR Centered Tree (GRCT) is shown in Figure 2, where the POS-Tags and lexemes are children of GR nodes. (ii) The second type of tree uses lexicals as central nodes on which both GR and POS-Tag are added as the right-most children. For example, Figure 3 shows an example of a Lexical Centered Tree (LCT). For both trees, the pruning strategy only preserves the verb node, its direct ancestors (father and siblings) and its descendants up to two levels (i.e., direct children and grandchildren of the verb node). Note that, our dependency tree can capture the semantic head of the verbal argument along with the main syntactic construct, e.g., *to audit*.

#### 4.2 Generalized node similarity for SPTK

We have defined the new similarity  $\sigma_\tau$  to be used in Eq. 2, which makes SPTK more effective as shown by Alg. 1.  $\sigma_\tau$  takes two nodes  $n_1$  and  $n_2$  and applies a different similarity for each node type. The latter is derived by  $\tau$  and can be: GR (i.e., SYNT), POS-Tag (i.e., POS) or a lexical (i.e., LEX) type. In our experiment, we assign 0/1 similarity for SYNT and POS nodes according to string matching. For LEX type, we apply a lexical similarity learned with LSA to only pairs of lexicals associated with the same POS-Tag. It should be noted that the type-based similarity allows for potentially applying a different similarity for each node. Indeed, we also tested an amplification factor, namely, leaf weight ( $lw$ ), which amplifies the matching values of the leaf nodes.

---

#### Algorithm 1 $\sigma_\tau(n_1, n_2, lw)$

---

```

 $\sigma_\tau \leftarrow 0$ ,
if  $\tau(n_1) = \tau(n_2) = \text{SYNT} \wedge \text{label}(n_1) = \text{label}(n_2)$  then
   $\sigma_\tau \leftarrow 1$ 
end if
if  $\tau(n_1) = \tau(n_2) = \text{POS} \wedge \text{label}(n_1) = \text{label}(n_2)$  then
   $\sigma_\tau \leftarrow 1$ 
end if
if  $\tau(n_1) = \tau(n_2) = \text{LEX} \wedge \text{pos}(n_1) = \text{pos}(n_2)$  then
   $\sigma_\tau \leftarrow \sigma_{\text{LEX}}(n_1, n_2)$ 
end if
if  $\text{leaf}(n_1) \wedge \text{leaf}(n_2)$  then
   $\sigma_\tau \leftarrow \sigma_\tau \times lw$ 
end if
return  $\sigma_\tau$ 

```

---

## 5 Experiments

In these experiments, we tested the impact of our different verb representations using different kernels, similarities and parameters. We also compared with simple bag-of-words (BOW) models and the state-of-the-art.

### 5.1 General experimental setup

We consider two different corpora: one for VerbNet and the other for FrameNet. For the former, we used the same verb classification setting of (Brown et al., 2011). Sentences are drawn from the Semlink corpus (Loper et al., 2007), which consists of the Prop-Banked Penn Treebank portions of the Wall Street Journal. It contains 113K verb instances, 97K of which are verbs represented in at least one VerbNet class. Semlink includes 495 verbs, whose instances are labeled with more than one class (including one single VerbNet class or none). We used all instances of the corpus for a total of 45,584 instances for 180 verb classes. When instances labeled with the *none* class are not included, the number of examples becomes 23,719.

The second corpus refers to FrameNet frame classification. The training and test data are drawn from the FrameNet 1.5 corpus<sup>2</sup>, which consists of 135K sentences annotated according the frame semantics

<sup>2</sup><http://framenet.icsi.berkeley.edu>

(Baker et al., 1998). We selected the subset of frames containing more than 100 sentences annotated with a verbal predicate for a total of 62,813 sentences in 187 frames (i.e., very close to the VerbNet datasets). For both the datasets, we used 70% of instances for training and 30% for testing.

Our verb (multi) classifier is designed with the *one-vs-all* (Rifkin and Klautau, 2004) multi-classification schema. This uses a set of binary SVM classifiers, one for each verb class (frame)  $i$ . The sentences whose verb is labeled with the class  $i$  are positive examples for the classifier  $i$ . The sentences whose verbs are compatible with the class  $i$  but evoking a different class or labeled with *none* (no current verb class applies) are added as negative examples. In the classification phase the binary classifiers are applied by (i) only considering classes that are compatible with the target verbs; and (ii) selecting the class associated with the maximum positive SVM margin. If all classifiers provide a negative score the example is labeled with *none*.

To learn the binary classifiers of the schema above, we coded our modified SPTK in SVM-Light-TK<sup>3</sup> (Moschitti, 2006). The parameterization of each classifier is carried on a held-out set (30% of the training) and is concerned with the setting of the trade-off parameter (option -c) and the leaf weight ( $lw$ ) (see Alg. 1), which is used to linearly scale the contribution of the leaf nodes. In contrast, the cost-factor parameter of SVM-Light-TK is set as the ratio between the number of negative and positive examples for attempting to have a balanced Precision/Recall.

Regarding SPTK setting, we used the lexical similarity  $\sigma$  defined in Sec. 3.1. In more detail, LSA was applied to ukWak (Baroni et al., 2009), which is a large scale document collection made up of 2 billion tokens.  $M$  is constructed by applying POS tagging to build rows with pairs  $\langle \text{lemma}, ::\text{POS} \rangle$  (lemma::POS in brief). The contexts of such items are the columns of  $M$  and are short windows of size  $[-3, +3]$ , centered on the items. This allows for better capturing syntactic properties of words. The most frequent 20,000 items are selected along with their 20k contexts. The entries of  $M$  are the point-wise mutual

<sup>3</sup>(Structural kernels in SVMLight (Joachims, 2000)) available at <http://disi.unitn.it/moschitti/Tree-Kernel.htm>

	STK		PTK		SPTK	
	$lw$	Acc.	$lw$	Acc.	$lw$	Acc.
CT	-	83.83%	8	<b>84.57%</b>	8	84.46%
GRCT	-	84.83%	8	85.15%	8	<b>85.28%</b>
LCT	-	77.73%	0.1	86.03%	0.2	<b>86.72%</b>
Br. et Al.	84.64%					
BOW	79.08%					
SK	82.08%					

Table 1: VerbNet accuracy with the *none* class

	STK		PTK		SPTK	
	$lw$	Acc.	$lw$	Acc.	$lw$	Acc.
GRCT	-	92.67%	6	92.97%	0.4	93.54%
LCT	-	90.28%	6	92.99%	0.3	93.78%
BOW	91.13%					
SK	91.84%					

Table 2: FrameNet accuracy without the *none* class

information between them. SVD reduction is then applied to  $M$ , with a dimensionality cut of  $l = 250$ .

For generating the CT, GRCT and LCT structures, we used the constituency trees generated by the Charniak parser (Charniak, 2000) and the dependency structures generated by the LTH syntactic parser (described in (Johansson and Nugues, 2008)).

The classification performance is measured with accuracy (i.e., the percentage of correct classification). We also derive statistical significance of the results by using the model described in (Yeh, 2000) and implemented in (Padó, 2006).

## 5.2 VerbNet and FrameNet Classification Results

To assess the performance of our settings, we also derive a simple baseline based on the bag-of-words (BOW) model. For it, we represent an instance of a verb in a sentence using all words of the sentence (by creating a special feature for the predicate word).

We also used sequence kernels (SK), i.e., PTK applied to a tree composed of a fake root and only one level of sentence words. For efficiency reasons<sup>4</sup>, we only consider the 10 words before and after the predicate with subsequence features of length up to 5.

Table 1 reports the accuracy of different models for VerbNet classification. It should be noted that: first, SK produces a much higher accuracy than BOW, i.e., 82.08 vs. 79.08. On one hand, this is

<sup>4</sup>The average running time of the SK is much higher than the one of PTK. When a tree is composed by only one level PTK collapses to SK.



	STK		PTK		SPTK	
	<i>lw</i>	Acc.	<i>lw</i>	Acc.	<i>lw</i>	Acc.
CT	-	91.14%	8	91.66%	6	91.66%
GRCT	-	91.71%	8	92.38%	4	92.33%
LCT	-	89.20%	0.2	92.54%	0.1	92.55%
BOW				88.16%		
SK				89.86%		

Table 3: VerbNet accuracy without the *none* class

generally in contrast with standard text categorization tasks, for which n-gram models show accuracy comparable to the simpler BOW. On the other hand, it simply confirms that verb classification requires the dependency information between words (i.e., at least the sequential structure information provided by SK).

Second, SK is 2.56 percent points below the state-of-the-art achieved in (Brown et al., 2011) (BR), i.e., 82.08 vs. 84.64. In contrast, STK applied to our representation (CT, GRCT and LCT) produces comparable accuracy, e.g., 84.83, confirming that syntactic representation is needed to reach the state-of-the-art.

Third, PTK, which produces more general structures, improves over BR by almost 1.5 (statistically significant result) when using our dependency structures GRCT and LCT. CT does not produce the same improvement since it does not allow PTK to directly compare the lexical structure (lexemes are all leaf nodes in CT and to connect some pairs of them very large trees are needed).

Finally, the best model of SPTK (i.e., using LCT) improves over the best PTK (i.e., using LCT) by almost 1 point (statistically significant result): this difference is only given by lexical similarity. SPTK improves on the state-of-the-art by about 2.08 absolute percent points, which, given the high accuracy of the baseline, corresponds to 13.5% of relative error reduction.

We carried out similar experiments for frame classification. One interesting difference is that SK improves BOW by only 0.70, i.e., 4 times less than in the VerbNet setting. This suggests that word order around the predicate is more important for deriving the VerbNet class than the FrameNet frame. Additionally, LCT or GRCT seems to be invariant for both PTK and SPTK whereas the lexical similarity still produces a relevant improvement on PTK, i.e., 13% of relative error reduction, for an absolute accuracy of 93.78%. The latter improves over the state-

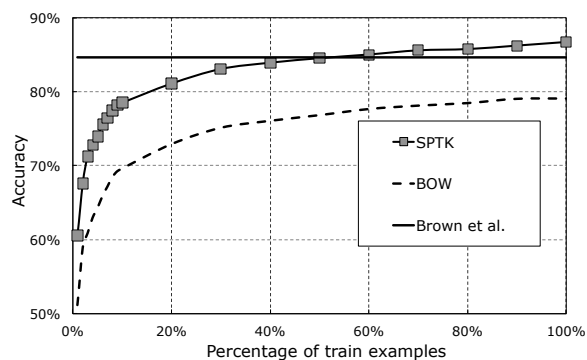


Figure 4: Learning curves: VerbNet accuracy with the *none* Class

of-the-art, i.e., 92.63% derived in (Giuglea and Moschitti, 2006), by using STK on CT on 133 frames.

We also carried out experiments to understand the role of the *none* class. Table 3 reports on the VerbNet classification without its instances. This is of course an unrealistic setting as it would assume that the current VerbNet release already includes all senses for English verbs. In the table, we note that the overall accuracy highly increases and the difference between models reduces. The similarities play no role anymore. This may suggest that SPTK can help in complex settings, where verb class characterization is more difficult. Another important role of SPTK models is their ability to generalize. To test this aspect, Figure 4 illustrates the learning curves of SPTK with respect to BOW and the accuracy achieved by BR (with a constant line). It is impressive to note that with only 40% of the data SPTK can reach the state-of-the-art.

## 6 Model Analysis and Discussion

We carried out analysis of system errors and its induced features. These can be examined by applying the reverse engineering tool<sup>5</sup> proposed in (Pighin and Moschitti, 2010; Pighin and Moschitti, 2009a; Pighin and Moschitti, 2009b), which extracts the most important features for the classification model. Many mistakes are related to false positives and negatives of the *none* class (about 72% of the errors). This class also causes data imbalance. Most errors are also due to lack of lexical information available to the SPTK kernel: (i) in 30% of the errors, the argument heads were proper nouns for which the lexical generalization provided by the DMs was not

<sup>5</sup><http://danielepighin.net/cms/software/flink>

VerbNet class 13.5.1
(IM(VB(target)))(OBJ)
(VC(VB(target)))(OBJ)
(VC(VBG(target)))(OBJ)
(OPRD(TO)(IM(VB(target)))(OBJ))
(PMOD(VBG(target)))(OBJ)
(VB(target))
(VC(VBN(target)))
(PRP(TO)(IM(VB(target)))(OBJ))
(IM(VB(target)))(OBJ)(ADV(IN)(PMOD)))
(OPRD(TO)(IM(VB(target)))(OBJ)(ADV(IN)(PMOD)))
VerbNet class 60
(VC(VB(target)))(OBJ)
(NMOD(VBG(target)))(OPRD)
(VC(VBN(target)))(OPRD)
(NMOD(VBN(target)))(OPRD)
(PMOD(VBG(target)))(OBJ)
(ROOT(SBJ)(VBD(target)))(OBJ)(P(.))
(VC(VB(target)))(OPRD)
(ROOT(SBJ)(VBZ(target)))(OBJ)(P(.))
(NMOD(SBJ)(WDT))(VBZ(target)))(OPRD)
(NMOD(SBJ)(VBZ(target)))(OPRD(SBJ)(TO)(IM))

Table 4: GRCT fragments

available; and (ii) in 76% of the errors only 2 or less argument heads are included in the extracted tree, therefore tree kernels cannot exploit enough lexical information to disambiguate verb senses. Additionally, ambiguity characterizes errors where the system is linguistically consistent but the learned selectional preferences are not sufficient to separate verb senses. These errors are mainly due to the lack of contextual information. While error analysis suggests that further improvement is possible (e.g. by exploiting proper nouns), the type of generalizations currently achieved by SPTK are rather effective. Table 4 and 5 report the tree structures characterizing the most informative training examples of the two senses of the verb *order*, i.e. the VerbNet classes 13.5.1 (*make a request for something*) and 60 (*give instructions to or direct somebody to do something with authority*).

In line with the method discussed in (Pighin and Moschitti, 2009b), these fragments are extracted as they appear in most of the support vectors selected during SVM training. As easily seen, the two classes are captured by rather different patterns. The typical accusative form with an explicit direct object emerges as characterizing the sense 13.5.1, denoting the THEME role. All fragments of the sense 60 emphasize instead the sentential complement of the verb that in fact expresses the standard PROPOSITION role in VerbNet. Notice that tree fragments correspond to syntactic patterns. The a posteriori

VerbNet class 13.5.1
(VP(VB(target)))(NP)
(VP(VBG(target)))(NP)
(VP(VBD(target)))(NP)
(VP(TO)(VP(VB(target)))(NP))
(S(NP-SBJ)(VP(VBP(target)))(NP))
VerbNet class 60
(VBN(target))
(VP(VBD(target)))(S)
(VP(VBZ(target)))(S)
(VBP(target))
(VP(VBD(target)))(NP-1)(S(NP-SBJ)(VP))

Table 5: CT fragments

analysis of the learned models (i.e. the underlying support vectors) confirm very interesting grammatical generalizations, i.e. the capability of tree kernels to implicitly trigger useful linguistic inductions for complex semantic tasks. When SPTK are adopted, verb arguments can be lexically generalized into word classes, i.e., clusters of argument heads (e.g. *commission* vs. *delegation*, or *gift* vs. *present*). Automatic generation of such classes is an interesting direction for future research.

## 7 Conclusion

We have proposed new approaches to characterize verb classes in learning algorithms. The key idea is the use of structural representation of verbs based on syntactic dependencies and the use of structural kernels to measure similarity between such representations. The advantage of kernel methods is that they can be directly used in some learning algorithms, e.g., SVMs, to train verb classifiers. Very interestingly, we can encode distributional lexical similarity in the similarity function acting over syntactic structures and this allows for generalizing selection restrictions through a *sort of* (supervised) syntactic and semantic co-clustering.

The verb classification results show a large improvement over the state-of-the-art for both VerbNet and FrameNet, with a relative error reduction of about 13.5% and 16.0%, respectively. In the future, we plan to exploit the models learned from FrameNet and VerbNet to carry out automatic mapping of verbs from one theory to the other.

**Acknowledgements** This research is partially supported by the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant numbers 247758 (ETERNALS), 288024 (LIMOSINE) and 231126 (LIVINGKNOWLEDGE). Many thanks to the reviewers for their valuable suggestions.

## References

- Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project.
- Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *LRE*, 43(3):209–226.
- Stephan Bloehdorn and Alessandro Moschitti. 2007a. Combined syntactic and semantic kernels for text classification. In Gianni Amati, Claudio Carpineto, and Gianni Romano, editors, *Proceedings of ECIR*, volume 4425 of *Lecture Notes in Computer Science*, pages 307–318. Springer, APR.
- Stephan Bloehdorn and Alessandro Moschitti. 2007b. Structure and semantics for expressive text kernels. In *CIKM'07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 861–864, New York, NY, USA. ACM.
- Susan Windisch Brown, Dmitriy Dligach, and Martha Palmer. 2011. Verbnet class assignment as a wsd task. In *Proceedings of the Ninth International Conference on Computational Semantics, IWCS '11*, pages 85–94, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of HLT and EMNLP*, pages 724–731, Vancouver, British Columbia, Canada, October.
- Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. 2003. Word sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL'00*.
- Michael Collins and Nigel Duffy. 2002. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron. In *Proceedings of ACL'02*.
- Nello Cristianini, John Shawe-Taylor, and Huma Lodhi. 2001. Latent semantic kernels. In Carla Brodley and Andrea Danyluk, editors, *Proceedings of ICML-01, 18th International Conference on Machine Learning*, pages 66–73, Williams College, US. Morgan Kaufmann Publishers, San Francisco, US.
- Danilo Croce, Cristina Giannone, Paolo Annesi, and Roberto Basili. 2010. Towards open-domain semantic role labeling. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 237–246, Uppsala, Sweden, July. Association for Computational Linguistics.
- Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured Lexical Similarity via Convolution Kernels on Dependency Trees. In *Proceedings of EMNLP 2011*.
- Chad Cumby and Dan Roth. 2003. Kernel Methods for Relational Learning. In *Proceedings of ICML 2003*.
- Hal Daumé III and Daniel Marcu. 2004. Np bracketing by maximum entropy tagging and SVM reranking. In *Proceedings of EMNLP'04*.
- Dmitriy Dligach and Martha Palmer. 2008. Novel semantic features for verb sense disambiguation. In *ACL (Short Papers)*, pages 29–32. The Association for Computer Linguistics.
- J. Firth. 1957. A synopsis of linguistic theory 1930-1955. In *Studies in Linguistic Analysis*. Philological Society, Oxford. reprinted in Palmer, F. (ed. 1968) *Selected Papers of J. R. Firth*, Longman, Harlow.
- G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, and K. E. Lochbaum. 1988. Information retrieval using a singular value decomposition model of latent semantic structure. In *Proc. of SIGIR '88*, New York, USA.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistic*, 28(3):496–530.
- Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02)*, Philadelphia, PA.
- Ana-Maria Giuglea and Alessandro Moschitti. 2006. Semantic role labeling via framenet, verbnet and propbank. In *Proceedings of ACL*, pages 929–936, Sydney, Australia, July.
- Alfio Gliozzo, Claudio Giuliano, and Carlo Strapparava. 2005. Domain kernels for word sense disambiguation. In *Proceedings of ACL'05*, pages 403–410.
- G. Golub and W. Kahan. 1965. Calculating the singular values and pseudo-inverse of a matrix. *Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis*.
- T. Joachims. 2000. Estimating the generalization performance of a SVM efficiently. In *Proceedings of ICML'00*.
- Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic-semantic analysis with PropBank and NomBank. In *Proceedings of CoNLL 2008*, pages 183–187.
- Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proceedings of ACL'03*.
- Taku Kudo, Jun Suzuki, and Hideki Isozaki. 2005. Boosting-based parse reranking with subtree features. In *Proceedings of ACL'05*.
- Tom Landauer and Sue Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory

- of acquisition, induction and representation of knowledge. *Psychological Review*, 104.
- Dekang Lin. 1998. Automatic retrieval and clustering of similar word. In *Proceedings of COLING-ACL*, Montreal, Canada.
- Edward Loper, Szu ting Yi, and Martha Palmer. 2007. Combining lexical resources: Mapping between propbank and verbnets. In *Proceedings of the 7th International Workshop on Computational Linguistics*.
- Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. 2010. Syntactic/semantic structures for textual entailment recognition. In *HLT-NAACL*, pages 1020–1028.
- Alessandro Moschitti. 2006. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of ECML'06*, pages 318–329.
- Sebastian Padó, 2006. *User's guide to sigf: Significance testing by approximate randomisation*.
- Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. Isp: Learning inferential selectional preferences. In *Proceedings of HLT/NAACL 2007*.
- Daniele Pighin and Alessandro Moschitti. 2009a. Efficient linearization of tree kernel functions. In *Proceedings of CoNLL'09*.
- Daniele Pighin and Alessandro Moschitti. 2009b. Reverse engineering of tree kernel feature spaces. In *Proceedings of EMNLP*, pages 111–120, Singapore, August. Association for Computational Linguistics.
- Daniele Pighin and Alessandro Moschitti. 2010. On reverse feature engineering of syntactic tree kernels. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL '10, pages 223–233, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Sameer Pradhan, Kadri Hacioglu, Valeri Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. *Machine Learning Journal*.
- Ryan Rifkin and Aldebaro Klautau. 2004. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141.
- Magnus Sahlgren. 2006. *The Word-Space Model*. Ph.D. thesis, Stockholm University.
- Karin Kipper Schuler. 2005. *VerbNet: A broad-coverage, comprehensive verb lexicon*. Ph.D. thesis, University of Pennsylvania.
- Hinrich Schutze. 1998. Automatic word sense discrimination. *Journal of Computational Linguistics*, 24:97–123.
- John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.
- Libin Shen, Anoop Sarkar, and Aravind k. Joshi. 2003. Using LTAG Based Features in Parse Reranking. In *Empirical Methods for Natural Language Processing (EMNLP)*, pages 89–96, Sapporo, Japan.
- Ivan Titov and James Henderson. 2006. Porting statistical parsers with data-defined kernels. In *Proceedings of CoNLL-X*.
- Kristina Toutanova, Penka Markova, and Christopher Manning. 2004. The Leaf Path Projection View of Parse Trees: Exploring String Kernels for HPSG Parse Selection. In *Proceedings of EMNLP 2004*.
- Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.
- Ludwig Wittgenstein. 1953. *Philosophical Investigations*. Blackwells, Oxford.
- Alexander S. Yeh. 2000. More accurate tests for the statistical significance of result differences. In *COLING*, pages 947–953.
- Beñat Zepirain, Eneko Agirre, Lluís Màrquez, and Mihai Surdeanu. 2010. Improving semantic role classification with selectional preferences. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 373–376, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of EMNLP-ACL*, pages 181–201.
- Min Zhang, Jie Zhang, and Jian Su. 2006. Exploring Syntactic Features for Relation Extraction using a Convolution tree kernel. In *Proceedings of NAACL*.

# Word Sense Disambiguation Improves Information Retrieval

Zhi Zhong and Hwee Tou Ng  
Department of Computer Science  
National University of Singapore  
13 Computing Drive, Singapore 117417  
{zhongzhi, nght}@comp.nus.edu.sg

## Abstract

Previous research has conflicting conclusions on whether word sense disambiguation (WSD) systems can improve information retrieval (IR) performance. In this paper, we propose a method to estimate sense distributions for short queries. Together with the senses predicted for words in documents, we propose a novel approach to incorporate word senses into the language modeling approach to IR and also exploit the integration of synonym relations. Our experimental results on standard *TREC* collections show that using the word senses tagged by a supervised WSD system, we obtain significant improvements over a state-of-the-art IR system.

## 1 Introduction

Word sense disambiguation (WSD) is the task of identifying the correct meaning of a word in context. As a basic semantic understanding task at the lexical level, WSD is a fundamental problem in natural language processing. It can be potentially used as a component in many applications, such as machine translation (MT) and information retrieval (IR).

In recent years, driven by Senseval/Semeval workshops, WSD systems achieve promising performance. In the application of WSD to MT, research has shown that integrating WSD in appropriate ways significantly improves the performance of MT systems (Chan et al., 2007; Carpuat and Wu, 2007).

In the application to IR, WSD can bring two kinds of benefits. First, queries may contain ambiguous words (terms), which have multiple meanings. The

ambiguities of these query words can hurt retrieval precision. Identifying the correct meaning of the ambiguous words in both queries and documents can help improve retrieval precision. Second, query words may have tightly related meanings with other words not in the query. Making use of these relations between words can improve retrieval recall.

Overall, IR systems can potentially benefit from the correct meanings of words provided by WSD systems. However, in previous investigations of the usage of WSD in IR, different researchers arrived at conflicting observations and conclusions. Some of the early research showed a drop in retrieval performance by using word senses (Krovetz and Croft, 1992; Voorhees, 1993). Some other experiments observed improvements by integrating word senses in IR systems (Schütze and Pedersen, 1995; Gonzalo et al., 1998; Stokoe et al., 2003; Kim et al., 2004).

This paper proposes the use of word senses to improve the performance of IR. We propose an approach to annotate the senses for short queries. We incorporate word senses into the language modeling (LM) approach to IR (Ponte and Croft, 1998), and utilize sense synonym relations to further improve the performance. Our evaluation on standard *TREC*<sup>1</sup> data sets shows that supervised WSD outperforms two other WSD baselines and significantly improves IR.

The rest of this paper is organized as follows. In Section 2, we first review previous work using WSD in IR. Section 3 introduces the LM approach to IR, including the pseudo relevance feedback method. We describe our WSD system and the method of

<sup>1</sup><http://trec.nist.gov/>

generating word senses for query terms in Section 4, followed by presenting our novel method of incorporating word senses and their synonyms into the LM approach in Section 5. We present experiments and analyze the results in Section 6. Finally, we conclude in Section 7.

## 2 Related Work

Many previous studies have analyzed the benefits and the problems of applying WSD to IR. Krovetz and Croft (1992) studied the sense matches between terms in query and the document collection. They concluded that the benefits of WSD in IR are not as expected because query words have skewed sense distribution and the collocation effect from other query terms already performs some disambiguation. Sanderson (1994; 2000) used pseudowords to introduce artificial word ambiguity in order to study the impact of sense ambiguity on IR. He concluded that because the effectiveness of WSD can be negated by inaccurate WSD performance, high accuracy of WSD is an essential requirement to achieve improvement. In another work, Gonzalo *et al.* (1998) used a manually sense annotated corpus, SemCor, to study the effects of incorrect disambiguation. They obtained significant improvements by representing documents and queries with accurate senses as well as synsets (synonym sets). Their experiment also showed that with the synset representation, which included synonym information, WSD with an error rate of 40%–50% can still improve IR performance. Their later work (Gonzalo *et al.*, 1999) verified that part of speech (POS) information is discriminatory for IR purposes.

Several works attempted to disambiguate terms in both queries and documents with the senses predefined in hand-crafted sense inventories, and then used the senses to perform indexing and retrieval. Voorhees (1993) used the hyponymy (“IS-A”) relation in WordNet (Miller, 1990) to disambiguate the polysemous nouns in a text. In her experiments, the performance of sense-based retrieval is worse than stem-based retrieval on all test collections. Her analysis showed that inaccurate WSD caused the poor results.

Stokoe *et al.* (2003) employed a fine-grained WSD system with an accuracy of 62.1% to dis-

ambiguate terms in both the text collections and the queries in their experiments. Their evaluation on TREC collections achieved significant improvements over a standard term based vector space model. However, it is hard to judge the effect of word senses because of the overall poor performances of their baseline method and their system.

Instead of using fine-grained sense inventory, Kim *et al.* (2004) tagged words with 25 root senses of nouns in WordNet. Their retrieval method maintained the stem-based index and adjusted the term weight in a document according to its sense matching result with the query. They attributed the improvement achieved on TREC collections to their coarse-grained, consistent, and flexible sense tagging method. The integration of senses into the traditional stem-based index overcomes some of the negative impact of disambiguation errors.

Different from using predefined sense inventories, Schütze and Pedersen (1995) induced the sense inventory directly from the text retrieval collection. For each word, its occurrences were clustered into senses based on the similarities of their contexts. Their experiments showed that using senses improved retrieval performance, and the combination of word-based ranking and sense-based ranking can further improve performance. However, the clustering process of each word is a time consuming task. Because the sense inventory is collection dependent, it is also hard to expand the text collection without re-doing preprocessing.

Many studies investigated the expansion effects by using knowledge sources from thesauri. Some researchers achieved improvements by expanding the disambiguated query words with synonyms and some other information from WordNet (Voorhees, 1994; Liu *et al.*, 2004; Liu *et al.*, 2005; Fang, 2008). The usage of knowledge sources from WordNet in document expansion also showed improvements in IR systems (Cao *et al.*, 2005; Agirre *et al.*, 2010).

The previous work shows that the WSD errors can easily neutralize its positive effect. It is important to reduce the negative impact of erroneous disambiguation, and the integration of senses into traditional term index, such as stem-based index, is a possible solution. The utilization of semantic relations has proved to be helpful for IR. It is also interest-

ing to investigate the utilization of semantic relations among senses in IR.

### 3 The Language Modeling Approach to IR

This section describes the LM approach to IR and the pseudo relevance feedback approach.

#### 3.1 The language modeling approach

In the language modeling approach to IR, language models are constructed for each query  $q$  and each document  $d$  in a text collection  $C$ . The documents in  $C$  are ranked by the distance to a given query  $q$  according to the language models. The most commonly used language model in IR is the unigram model, in which terms are assumed to be independent of each other. In the rest of this paper, language model will refer to the unigram language model.

One of the commonly used measures of the similarity between query model and document model is negative Kullback-Leibler (KL) divergence (Lafferty and Zhai, 2001). With unigram model, the negative KL-divergence between model  $\theta_q$  of query  $q$  and model  $\theta_d$  of document  $d$  is calculated as follows:

$$\begin{aligned} -D(\theta_q||\theta_d) &= -\sum_{t \in V} p(t|\theta_q) \log \frac{p(t|\theta_q)}{p(t|\theta_d)} \\ &= \sum_{t \in V} p(t|\theta_q) \log p(t|\theta_d) - \sum_{t \in V} p(t|\theta_q) \log p(t|\theta_q) \\ &= \sum_{t \in V} p(t|\theta_q) \log p(t|\theta_d) + E(\theta_q), \end{aligned} \quad (1)$$

where  $p(t|\theta_q)$  and  $p(t|\theta_d)$  are the generative probabilities of a term  $t$  from the models  $\theta_q$  and  $\theta_d$ ,  $V$  is the vocabulary of  $C$ , and  $E(\theta_q)$  is the entropy of  $q$ .

Define  $tf(t, d)$  and  $tf(t, q)$  as the frequencies of  $t$  in  $d$  and  $q$ , respectively. Normally,  $p(t|\theta_q)$  is calculated with maximum likelihood estimation (MLE):

$$p(t|\theta_q) = \frac{tf(t, q)}{\sum_{t' \in q} tf(t', q)}. \quad (2)$$

In the calculation of  $p(t|\theta_d)$ , several smoothing methods have been proposed to overcome the data sparseness problem of a language model constructed from one document (Zhai and Lafferty, 2001b). For example,  $p(t|\theta_d)$  with the Dirichlet-prior smoothing can be calculated as follows:

$$p(t|\theta_d) = \frac{tf(t, d) + \mu p(t|\theta_C)}{\sum_{t' \in V} tf(t', d) + \mu}, \quad (3)$$

where  $\mu$  is the prior parameter in the Dirichlet-prior smoothing method, and  $p(t|\theta_C)$  is the probability of  $t$  in  $C$ , which is often calculated with MLE:

$$p(t|\theta_C) = \frac{\sum_{d' \in C} tf(t, d')}{\sum_{d' \in C} \sum_{t' \in V} tf(t', d')}.$$

#### 3.2 Pseudo relevance feedback

Pseudo relevance feedback (PRF) is widely used in IR to achieve better performance. It is constructed with two retrieval steps. In the first step, ranked documents are retrieved from  $C$  by a normal retrieval method with the original query  $q$ . In the second step, a number of terms are selected from the top  $k$  ranked documents  $D_q$  for query expansion, under the assumption that these  $k$  documents are relevant to the query. Then, the expanded query is used to retrieve the documents from  $C$ .

There are several methods to select expansion terms in the second step (Zhai and Lafferty, 2001a). For example, in Indri<sup>2</sup>, the terms are first ranked by the following score:

$$v(t, D_q) = \sum_{d \in D_q} \log \left( \frac{tf(t, d)}{|d|} \times \frac{1}{p(t|\theta_C)} \right),$$

as in Ponte (1998). Define  $p(q|\theta_d)$  as the probability score assigned to  $d$ . The top  $m$  terms  $T_q$  are selected with weights calculated based on the relevance model described in Lavrenko and Croft (2001):

$$w(t, D_q) = \sum_{d \in D_q} \left[ \frac{tf(t, d)}{|d|} \times p(q|\theta_d) \times p(\theta_d) \right],$$

which calculates the sum of weighted probabilities of  $t$  in each document. After normalization, the probability of  $t$  in  $\theta_q^r$  is calculated as follows:

$$p(t|\theta_q^r) = \frac{w(t, D_q)}{\sum_{t' \in T_q} w(t', D_q)}.$$

Finally, the relevance model is interpolated with the original query model:

$$p(t|\theta_q^{prf}) = \lambda p(t|\theta_q^r) + (1 - \lambda)p(t|\theta_q), \quad (4)$$

where parameter  $\lambda$  controls the amount of feedback. The new model  $\theta_q^{prf}$  is used to replace the original one  $\theta_q$  in Equation 1.

Collection enrichment (CE) (Kwok and Chan, 1998) is a technique to improve the quality of the feedback documents by making use of an external target text collection  $X$  in addition to the original target  $C$  in the first step of PRF. The usage of  $X$  is supposed to provide more relevant feedback documents and feedback query terms.

<sup>2</sup><http://lemurproject.org/indri/>

## 4 Word Sense Disambiguation

In this section, we first describe the construction of our WSD system. Then, we propose the method of assigning senses to query terms.

### 4.1 Word sense disambiguation system

Previous research shows that translations in another language can be used to disambiguate the meanings of words (Chan and Ng, 2005; Zhong and Ng, 2009). We construct our supervised WSD system directly from parallel corpora.

To generate the WSD training data, 7 parallel corpora were used, including *Chinese Treebank*, *FBIS Corpus*, *Hong Kong Hansards*, *Hong Kong Laws*, *Hong Kong News*, *Sinorama News Magazine*, and *Xinhua Newswire*. These corpora were already aligned at sentence level. We tokenized English texts with *Penn Treebank Tokenizer*, and performed word segmentation on Chinese texts. Then, word alignment was performed on the parallel corpora with the *GIZA++* software (Och and Ney, 2003).

For each English morphological root  $e$ , the English sentences containing its occurrences were extracted from the word aligned output of *GIZA++*, as well as the corresponding translations of these occurrences. To minimize noisy word alignment result, translations with no Chinese character were deleted, and we further removed a translation when it only appears once, or its frequency is less than 10 and also less than 1% of the frequency of  $e$ . Finally, only the most frequent 10 translations were kept for efficiency consideration.

The English part of the remaining occurrences were used as training data. Because multiple English words may have the same Chinese translation, to differentiate them, each Chinese translation is concatenated with the English morphological root to form a word sense. We employed a supervised WSD system, *IMS*<sup>3</sup>, to train the WSD models. *IMS* (Zhong and Ng, 2010) integrates multiple knowledge sources as features. We used MaxEnt as the machine learning algorithm. Finally, the system can disambiguate the words by assigning probabilities to different senses.

<sup>3</sup><http://nlp.comp.nus.edu.sg/software/ims>

### 4.2 Estimating sense distributions for query terms

In IR, both terms in queries and the text collection can be ambiguous. Hence, WSD is needed to disambiguate these ambiguous terms. In most cases, documents in a text collection are full articles. Therefore, a WSD system has sufficient context to disambiguate the words in the document. In contrast, queries are usually short, often with only two or three terms in a query. Short queries pose a challenge to WSD systems since there is insufficient context to disambiguate a term in a short query.

One possible solution to this problem is to find some text fragments that contain a query term. Suppose we already have a basic IR method which does not require any sense information, such as the stem-based LM approach. Similar to the PRF method, assuming that the top  $k$  documents retrieved by the basic method are relevant to the query, these  $k$  documents can be used to represent query  $q$  (Broder et al., 2007; Bendersky et al., 2010; He and Wu, 2011). We propose a method to estimate the sense probabilities of each query term of  $q$  from these top  $k$  retrieved documents.

Suppose the words in all documents of the text collection are disambiguated with a WSD system, and each word occurrence  $w$  in document  $d$  is assigned a vector of senses,  $S(w)$ . Define the probability of assigning sense  $s$  to  $w$  as  $p(w, s, d)$ . Given a query  $q$ , suppose  $D_q$  is the set of top  $k$  documents retrieved by the basic method, with the probability score  $p(q|\theta_d)$  assigned to  $d \in D_q$ .

---

```
Given a query term  $t \in q$ 
 $S(t, q) = \{\}$ 
 $sum = 0$ 
for each document  $d \in D_q$ 
  for each word occurrence  $w \in d$ , whose stem form is
  identical to the stem form of  $t$ 
    for each sense  $s \in S(w)$ 
       $S(t, q) = S(t, q) \cup \{s\}$ 
       $p(t, s, q) = p(t, s, q) + p(q|\theta_d) p(w, s, d)$ 
       $sum = sum + p(q|\theta_d) p(w, s, d)$ 
for each sense  $s \in S(t, q)$ 
   $p(t, s, q) = p(t, s, q) / sum$ 
Return  $S(t, q)$ , with probability  $p(t, s, q)$  for  $s \in S(t, q)$ 
```

---

Figure 1: Process of generating senses for query terms

Figure 1 shows the pseudocode of calculating the



sense distribution for a query term  $t$  in  $q$  with  $D_q$ , where  $S(t, q)$  is the set of senses assigned to  $t$  and  $p(t, s, q)$  is the probability of tagging  $t$  as sense  $s$ . Basically, we utilized the sense distribution of the words with the same stem form in  $D_q$  as a proxy to estimate the sense probabilities of a query term. The retrieval scores are used to weight the information from the corresponding retrieved documents in  $D_q$ .

## 5 Incorporating Senses into Language Modeling Approaches

In this section, we propose to incorporate senses into the LM approach to IR. Then, we describe the integration of sense synonym relations into our model.

### 5.1 Incorporating senses as smoothing

With the method described in Section 4.2, both the terms in queries and documents have been sense tagged. The next problem is to incorporate the sense information into the language modeling approach.

Suppose  $p(t, s, q)$  is the probability of tagging a query term  $t \in q$  as sense  $s$ , and  $p(w, s, d)$  is the probability of tagging a word occurrence  $w \in d$  as sense  $s$ . Given a query  $q$  and a document  $d$  in text collection  $C$ , we want to re-estimate the language models by making use of the sense information assigned to them.

Define the frequency of  $s$  in  $d$  as:

$$stf(s, d) = \sum_{w \in d} p(w, s, d),$$

and the frequency of  $s$  in  $C$  as:

$$stf(s, C) = \sum_{d \in C} stf(s, d).$$

Define the frequencies of sense set  $S$  in  $d$  and  $C$  as:

$$stf(S, d) = \sum_{s \in S} stf(s, d),$$

$$stf(S, C) = \sum_{s \in S} stf(s, C).$$

For a term  $t \in q$ , with senses  $S(t, q):\{s_1, \dots, s_n\}$ , suppose  $V:\{p(t, s_1, q), \dots, p(t, s_n, q)\}$  is the vector of probabilities assigned to the senses of  $t$  and  $W:\{stf(s_1, d), \dots, stf(s_n, d)\}$  is the vector of frequencies of  $S(t, q)$  in  $d$ . The function  $\cos(t, q, d)$  calculates the cosine similarity between vector  $V$  and vector  $W$ . Assume  $D$  is a set of documents in  $C$  which contain any sense in  $S(t, q)$ , we define function  $\overline{\cos}(t, q) = \sum_{d \in D} \cos(t, q, d) / |D|$ , which calculates the mean of the sense cosine similarities, and define function  $\Delta \cos(t, q, d) = \cos(t, q, d) -$

$\overline{\cos}(t, q)$ , which calculates the difference between  $\cos(t, q, d)$  and the corresponding mean value.

Given a query  $q$ , we re-estimate the term frequency of query term  $t$  in  $d$  with sense information integrated as smoothing:

$$tf_{sen}(t, d) = tf(t, d) + sen(t, q, d), \quad (5)$$

where function  $sen(t, q, d)$  is a measure of  $t$ 's sense information in  $d$ , which is defined as follows:

$$sen(t, q, d) = \alpha^{\Delta \cos(t, q, d)} stf(S(t, q), d). \quad (6)$$

In  $sen(t, q, d)$ , the last item  $stf(S(t, q), d)$  calculates the sum of the sense frequencies of  $t$  senses in  $d$ , which represents the amount of  $t$ 's sense information in  $d$ . The first item  $\alpha^{\Delta \cos(t, q, d)}$  is a weight of the sense information concerning the relative sense similarity  $\Delta \cos(t, q, d)$ , where  $\alpha$  is a positive parameter to control the impact of sense similarity. When  $\Delta \cos(t, q, d)$  is larger than zero, such that the sense similarity of  $d$  and  $q$  according to  $t$  is above the average, the weight for the sense information is larger than 1; otherwise, it is less than 1. The more similar they are, the larger the weight value. For  $t \notin q$ , because the sense set  $S(t, q)$  is empty,  $stf(S(t, q), d)$  equals to zero and  $tf_{sen}(t, d)$  is identical to  $tf(t, d)$ .

With sense incorporated, the term frequency is influenced by the sense information. Consequently, the estimation of probability of  $t$  in  $d$  becomes query specific:

$$p(t|\theta_d^{sen}) = \frac{tf_{sen}(t, d) + \mu p(t|\theta_C^{sen})}{\sum_{t' \in V} tf_{sen}(t', d) + \mu}, \quad (7)$$

where the probability of  $t$  in  $C$  is re-calculated as:

$$p(t|\theta_C^{sen}) = \frac{\sum_{d' \in C} tf_{sen}(t, d')}{\sum_{d' \in C} \sum_{t' \in V} tf_{sen}(t', d')}.$$

### 5.2 Expanding with synonym relations

Words usually have some semantic relations with others. Synonym relation is one of the semantic relations commonly used to improve IR performance. In this part, we further integrate the synonym relations of senses into the LM approach.

Suppose  $R(s)$  is the set of senses having synonym relation with sense  $s$ . Define  $S(q)$  as the set of senses of query  $q$ ,  $S(q) = \bigcup_{t \in q} S(t, q)$ , and define  $R(s, q) = R(s) - S(q)$ . We update the frequency of a query term  $t$  in  $d$  by integrating the synonym relations as follows:

$$tf_{syn}(t, d) = tf_{sen}(t, d) + syn(t, q, d), \quad (8)$$

where  $syn(t, q, d)$  is a function measuring the synonym information in  $d$ :

$$syn(t, q, d) = \sum_{s \in S(t)} \beta(s, q) p(t, s, q) stf(R(s, q), d).$$

The last item  $stf(R(s, q), d)$  in  $syn(t, q, d)$  is the sum of the sense frequencies of  $R(s, q)$  in  $d$ . Notice that the synonym senses already appearing in  $S(q)$  are not included in the calculation, because the information of these senses has been used in some other places in the retrieval function. The frequency of synonyms,  $stf(R(s, q), d)$ , is weighted by  $p(t, s, q)$  together with a scaling function  $\beta(s, q)$ :

$$\beta(s, q) = \min(1, \frac{stf(s, C)}{stf(R(s, q), C)}).$$

When  $stf(s, C)$ , the frequency of sense  $s$  in  $C$ , is less than  $stf(R(s, q), C)$ , the frequency of  $R(s, q)$  in  $C$ , the function  $\beta(s, q)$  scales down the impact of synonyms according to the ratio of these two frequencies. The scaling function makes sure that the overall impact of the synonym senses is not greater than the original word senses.

Accordingly, we have the probability of  $t$  in  $d$  updated to:

$$p(t|\theta_d^{syn}) = \frac{tf_{syn}(t, d) + \mu p(t|\theta_C^{syn})}{\sum_{t' \in V} tf_{syn}(t', d) + \mu}, \quad (9)$$

and the probability of  $t$  in  $C$  is calculated as:

$$p(t|\theta_C^{syn}) = \frac{\sum_{d' \in C} tf_{syn}(t, d')}{\sum_{d' \in C} \sum_{t' \in V} tf_{syn}(t', d')}.$$

With this language model, the probability of a query term in a document is enlarged by the synonyms of its senses; The more its synonym senses in a document, the higher the probability. Consequently, documents with more synonym senses of the query terms will get higher retrieval rankings.

## 6 Experiments

In this section, we evaluate and analyze the models proposed in Section 5 on standard TREC collections.

### 6.1 Experimental settings

We conduct experiments on the TREC collection. The text collection  $C$  includes the documents from TREC disk 4 and 5, minus the CR (Congressional Record) corpus, with 528,155 documents in total. In

addition, the other documents in TREC disk 1 to 5 are used as the external text collection  $X$ .

We use 50 queries from TREC6 Ad Hoc task as the development set, and evaluate on 50 queries from TREC7 Ad Hoc task, 50 queries from TREC8 Ad Hoc task, 50 queries from ROBUST 2003 (RB03), and 49 queries from ROBUST 2004 (RB04). In total, our test set includes 199 queries. We use the terms in the title field of TREC topics as queries. Table 1 shows the statistics of the five query sets. The first column lists the query topics, and the column *#qry* is the number of queries. The column *Ave* gives the average query length, and the column *Rels* is the total number of relevant documents.

Query Set	Topics	#qry	Ave	Rels
TREC6	301–350	50	2.58	4,290
TREC7	351–400	50	2.50	4,674
TREC8	401–450	50	2.46	4,728
RB03	601–650	50	3.00	1,658
RB04 <sup>4</sup>	651–700	49	2.96	2,062

Table 1: Statistics of query sets

We use the *Lemur* toolkit (Ogilvie and Callan, 2001) version 4.11 as the basic retrieval tool, and select the default unigram LM approach based on KL-divergence and Dirichlet-prior smoothing method in Lemur as our basic retrieval approach. Stop words are removed from queries and documents using the standard INQUERY stop words list (Allan et al., 2000), and then the Porter stemmer is applied to perform stemming. The stem forms are finally used for indexing and retrieval.

We set the smoothing parameter  $\mu$  in Equation 3 to 400 by tuning on TREC6 query set in a range of  $\{100, 400, 700, 1000, 1500, 2000, 3000, 4000, 5000\}$ . With this basic method, up to 10 top ranked documents  $D_q$  are retrieved for each query  $q$  from the extended text collection  $C \cup X$ , for the usage of performing PRF and generating query senses.

For PRF, we follow the implementation of Indri’s PRF method and further apply the CE technique as described in Section 3.2. The number of terms selected from  $D_q$  for expansion is tuned from range  $\{20, 25, 30, 35, 40\}$  and set to 25. The interpolation parameter  $\lambda$  in Equation 4 is set to 0.7 from range

<sup>4</sup>Topic 672 is eliminated, since it has no relevant document.

Method	TREC7	TERC8	RB03	RB04	Comb	Impr	#ret-rel
Top 1	0.2530	0.3063	0.3704	<b>0.4019</b>	-	-	-
Top 2	0.2488	0.2876	0.3065	0.4008	-	-	-
Top 3	0.2427	0.2853	0.3037	0.3514	-	-	-
Stem <sub>prf</sub> (Baseline)	0.2634	0.2944	0.3586	0.3781	0.3234	-	9248
Stem <sub>prf</sub> +MFS	0.2655	0.2971	0.3626 <sup>†</sup>	0.3802	0.3261 <sup>†</sup>	0.84%	9281
Stem <sub>prf</sub> +Even	0.2655	0.2972	0.3623 <sup>†</sup>	0.3814	0.3263 <sup>‡</sup>	0.91%	9284
Stem <sub>prf</sub> +WSD	0.2679 <sup>‡</sup>	0.2986 <sup>†</sup>	0.3649 <sup>‡</sup>	0.3842	0.3286 <sup>‡</sup>	1.63%	9332
Stem <sub>prf</sub> +MFS+Syn	0.2756 <sup>‡</sup>	0.3034 <sup>†</sup>	0.3649 <sup>†</sup>	0.3859	0.3322 <sup>‡</sup>	2.73%	9418
Stem <sub>prf</sub> +Even+Syn	0.2713 <sup>†</sup>	0.3061 <sup>‡</sup>	0.3657 <sup>‡</sup>	0.3859 <sup>†</sup>	0.3320 <sup>‡</sup>	2.67%	9445
Stem <sub>prf</sub> +WSD+Syn	<b>0.2762<sup>‡</sup></b>	<b>0.3126<sup>‡</sup></b>	<b>0.3735<sup>‡</sup></b>	0.3891 <sup>†</sup>	0.3376 <sup>‡</sup>	4.39%	9538

Table 2: Results on test set in MAP score. The first three rows show the results of the top participating systems, the next row shows the performance of the baseline method, and the rest rows are the results of our method with different settings. Single dagger (<sup>†</sup>) and double dagger (<sup>‡</sup>) indicate statistically significant improvement over *Stem<sub>prf</sub>* at the 95% and 99% confidence level with a two-tailed paired t-test, respectively. The best results are highlighted in bold.

{0.1, 0.2, ..., 0.9}. The CE-PRF method with this parameter setting is chosen as the baseline.

To estimate the sense distributions for terms in query  $q$ , the method described in Section 4.2 is applied with  $D_q$ . To disambiguate the documents in the text collection, besides the usage of the supervised WSD system described in Section 4.1, two WSD baseline methods, *Even* and *MFS*, are applied for comparison. The method *Even* assigns equal probabilities to all senses for each word, and the method *MFS* tags the words with their corresponding most frequent senses. The parameter  $\alpha$  in Equation 6 is tuned on *TREC6* from 1 to 10 in increment of 1 for each sense tagging method. It is set to 7, 6, and 9 for the supervised WSD method, the *Even* method, and the *MFS* method, respectively.

Notice that the sense in our WSD system is conducted with two parts, a morphological root and a Chinese translation. The Chinese parts not only disambiguate senses, but also provide clues of connections among different words. Assume that the senses with the same Chinese part are synonyms, therefore, we can generate a set of synonyms for each sense, and then utilize these synonym relations in the method proposed in Section 5.2.

## 6.2 Experimental results

For evaluation, we use average precision (AP) as the metric to evaluate the performance on each query  $q$ :

$$AP(q) = \frac{\sum_{r=1}^R [p(r)rel(r)]}{relevance(q)},$$

where  $relevance(q)$  is the number of documents relevant to  $q$ ,  $R$  is the number of retrieved documents,

$r$  is the rank,  $p(r)$  is the precision of the top  $r$  retrieved documents, and  $rel(r)$  equals to 1 if the  $r$ th document is relevant, and 0 otherwise. Mean average precision (MAP) is a metric to evaluate the performance on a set of queries  $Q$ :

$$MAP(Q) = \frac{\sum_{q \in Q} AP(q)}{|Q|},$$

where  $|Q|$  is the number of queries in  $Q$ .

We retrieve the top-ranked 1,000 documents for each query, and use the MAP score as the main comparing metric. In Table 2, the first four columns are the MAP scores of various methods on the TREC7, TREC8, RB03, and RB04 query sets, respectively. The column *Comb* shows the results on the union of the four test query sets. The first three rows list the results of the top three systems that participated in the corresponding tasks. The row *Stem<sub>prf</sub>* shows the performance of our baseline method, the stem-based CE-PRF method. The column *Impr* calculates the percentage improvement of each method over the baseline *Stem<sub>prf</sub>* in column *Comb*. The last column *#ret-rel* lists the total numbers of relevant documents retrieved by different methods.

The rows *Stem<sub>prf</sub>+{MFS, Even, WSD}* are the results of *Stem<sub>prf</sub>* incorporating with the senses generated for the original query terms, by applying the approach proposed in Section 5.1, with the *MFS* method, the *Even* method, and our supervised WSD method, respectively. Comparing to the baseline method, all methods with sense integrated achieve consistent improvements on all query sets. The usage of the supervised WSD method outperforms the other two WSD baselines, and it achieves sta-

tistically significant improvements over  $Stem_{prf}$  on TREC7, TREC8, and RB03.

The integration of senses into the baseline method has two aspects of impact. First, the morphological roots of senses conquer the irregular inflection problem. Thus, the documents containing the irregular inflections are retrieved when senses are integrated. For example, in topic 326 {*ferry sinkings*}, the stem form of *sinkings* is *sink*. As *sink* is an irregular verb, the usage of senses improves the retrieval recall by retrieving the documents containing the inflection forms *sunk*, *sank*, and *sunken*.

Second, the senses output by supervised WSD system help identify the meanings of query terms. Take topic 357 {*territorial waters dispute*} for example, the stem form of *waters* is *water* and its appropriate sense in this query should be water\_水域 (body of water) instead of the most frequent sense of water\_水 (H<sub>2</sub>O). In  $Stem_{prf}+WSD$ , we correctly identify the minority sense for this query term. In another example, topic 425 {*counterfeiting money*}, the stem form of *counterfeiting* is *counterfeit*. Although the most frequent sense counterfeit\_冒牌 (not genuine) is not wrong, another sense counterfeit\_伪钞 (forged money) is more accurate for this query term. The Chinese translation in the latter sense represents the meaning of the phrase in original query. Thus,  $Stem_{prf}+WSD$  outperforms the other two methods on this query by assigning the highest probability for this sense.

Overall, the performance of  $Stem_{prf}+WSD$  is better than  $Stem_{prf}+\{MFS, Even\}$  on 121 queries and 119 queries, respectively. The *t-test* at the confidence level of 99% indicates that the improvements are statistically significant.

The results of expanding with synonym relations in the above three methods are shown in the last three rows,  $Stem_{prf}+\{MFS, Even, WSD\}+Syn$ . The integration of synonym relations further improves the performance no matter what kind of sense tagging method is applied. The improvement varies with different methods on different query sets. As shown in the last column of Table 2, the number of relevant documents retrieved is increased for each method.  $Stem_{prf}+Even+Syn$  retrieves more relevant documents than  $Stem_{prf}+MFS+Syn$ , because the former method expands more senses. Overall, the improvement achieved by  $Stem_{prf}+WSD+Syn$  is

larger than the other two methods. It shows that the WSD technique can help choose the appropriate senses for synonym expansion.

Among the different settings,  $Stem_{prf}+WSD+Syn$  achieves the best performance. Its improvement over the baseline method is statistically significant at the 95% confidence level on RB04 and at the 99% confidence level on the other three query sets, with an overall improvement of 4.39%. It beats the best participated systems on three out of four query sets<sup>5</sup>, including *TREC7*, *TREC8*, and *RB03*.

## 7 Conclusion

This paper reports successful application of WSD to IR. We proposed a method for annotating senses to terms in short queries, and also described an approach to integrate senses into an LM approach for IR. In the experiment on four query sets of TREC collection, we compared the performance of a supervised WSD method and two WSD baseline methods. Our experimental results showed that the incorporation of senses improved a state-of-the-art baseline, a stem-based LM approach with PRF method. The performance of applying the supervised WSD method is better than the other two WSD baseline methods. We also proposed a method to further integrate the synonym relations to the LM approaches. With the integration of synonym relations, our best performance setting with the supervised WSD achieved an improvement of 4.39% over the baseline method, and it outperformed the best participating systems on three out of four query sets.

## Acknowledgments

This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

## References

- E. Agirre, X. Arregi, and A. Otegi. 2010. Document expansion based on WordNet for robust IR. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 9–17.

<sup>5</sup>The top two systems on *RB04* are the results of the same participant with different configurations. They used lots of web resources, such as search engines, to improve the performance.

- J. Allan, M. E. Connell, W.B. Croft, F.F. Feng, D. Fisher, and X. Li. 2000. INQUERY and TREC-9. In *Proceedings of the 9th Text REtrieval Conference*, pages 551–562.
- M. Bendersky, W. B. Croft, and D. A. Smith. 2010. Structural annotation of search queries using pseudo-relevance feedback. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, pages 1537–1540.
- A. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. 2007. Robust classification of rare queries using web knowledge. In *Proceedings of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 231–238.
- G. Cao, J. Y. Nie, and J. Bai. 2005. Integrating word relationships into language models. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 298–305.
- M. Carpuat and D. Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 61–72.
- Y. S. Chan and H. T. Ng. 2005. Scaling up word sense disambiguation via parallel texts. In *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 1037–1042.
- Y. S. Chan, H. T. Ng, and D. Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 33–40.
- H. Fang. 2008. A re-examination of query expansion using lexical resources. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technologies*, pages 139–147.
- J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarrin. 1998. Indexing with WordNet synsets can improve text retrieval. In *Proceedings of the COLING-ACL Workshop on Usage of WordNet in Natural Language Processing Systems*, pages 38–44.
- J. Gonzalo, A. Penas, and F. Verdejo. 1999. Lexical ambiguity and information retrieval revisited. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 195–202.
- D. He and D. Wu. 2011. Enhancing query translation with relevance feedback in translanguag information retrieval. *Information Processing & Management*, 47(1):1–17.
- S. B. Kim, H. C. Seo, and H. C. Rim. 2004. Information retrieval using word senses: root sense tagging approach. In *Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 258–265.
- R. Krovetz and W. B. Croft. 1992. Lexical ambiguity and information retrieval. *ACM Transactions on Information Systems*, 10(2):115–141.
- K. L. Kwok and M. Chan. 1998. Improving two-stage ad-hoc retrieval for short queries. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 250–256.
- J. Lafferty and C. Zhai. 2001. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 111–119.
- V. Lavrenko and W. B. Croft. 2001. Relevance based language models. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 120–127.
- S. Liu, F. Liu, C. Yu, and W. Meng. 2004. An effective approach to document retrieval via utilizing WordNet and recognizing phrases. In *Proceedings of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 266–272.
- S. Liu, C. Yu, and W. Meng. 2005. Word sense disambiguation in queries. In *Proceedings of the 14th ACM Conference on Information and Knowledge Management*, pages 525–532.
- G. A. Miller. 1990. WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–312.
- F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- P. Ogilvie and J. Callan. 2001. Experiments using the Lemur toolkit. In *Proceedings of the 10th Text REtrieval Conference*, pages 103–108.
- J. M. Ponte and W. B. Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of the 21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 275–281.
- J. M. Ponte. 1998. *A Language Modeling Approach to Information Retrieval*. Ph.D. thesis, Department of Computer Science, University of Massachusetts.
- M. Sanderson. 1994. Word sense disambiguation and information retrieval. In *Proceedings of the 17th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 142–151.

- M. Sanderson. 2000. Retrieving with good sense. *Information Retrieval*, 2(1):49–69.
- H. Schütze and J. O. Pedersen. 1995. Information retrieval based on word senses. In *Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175.
- C. Stokoe, M. P. Oakes, and J. Tait. 2003. Word sense disambiguation in information retrieval revisited. In *Proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 159–166.
- E. M. Voorhees. 1993. Using WordNet to disambiguate word senses for text retrieval. In *Proceedings of the 16th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 171–180.
- E. M. Voorhees. 1994. Query expansion using lexical-semantic relations. In *Proceedings of the 17th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 61–69.
- C. Zhai and J. Lafferty. 2001a. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the 10th ACM Conference on Information and Knowledge Management*, pages 403–410.
- C. Zhai and J. Lafferty. 2001b. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 334–342.
- Z. Zhong and H. T. Ng. 2009. Word sense disambiguation for all words without hard labor. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1616–1621.
- Z. Zhong and H. T. Ng. 2010. It Makes Sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the 48th Annual Meeting of the Association of Computational Linguistics: System Demonstrations*, pages 78–83.

# Efficient Search for Transformation-based Inference

Asher Stern<sup>§</sup>, Roni Stern<sup>‡</sup>, Ido Dagan<sup>§</sup>, Ariel Felner<sup>‡</sup>

<sup>§</sup> Computer Science Department, Bar-Ilan University

<sup>‡</sup> Information Systems Engineering, Ben Gurion University

astern7@gmail.com

roni.stern@gmail.com

dagan@cs.biu.ac.il

felner@bgu.ac.il

## Abstract

This paper addresses the search problem in textual inference, where systems need to infer one piece of text from another. A prominent approach to this task is attempts to transform one text into the other through a sequence of inference-preserving transformations, a.k.a. a proof, while estimating the proof's validity. This raises a search challenge of finding the best possible proof. We explore this challenge through a comprehensive investigation of prominent search algorithms and propose two novel algorithmic components specifically designed for textual inference: a gradient-style evaluation function, and a local-lookahead node expansion method. Evaluations, using the open-source system, BIUTEE, show the contribution of these ideas to search efficiency and proof quality.

## 1 Introduction

In many NLP settings it is necessary to identify that a certain semantic inference relation holds between two pieces of text. For example, in *paraphrase recognition* it is necessary to identify that the meanings of two text fragments are roughly equivalent. In *passage retrieval* for question answering, it is needed to detect text passages from which a satisfying answer can be inferred. A generic formulation for the inference relation between two texts is given by the *Recognizing Textual Entailment (RTE)* paradigm (Dagan et al., 2005), which is adapted here for our investigation. In this setting, a system is given two text fragments, termed “text” ( $T$ ) and “hy-

pothesis” ( $H$ ), and has to recognize whether the hypothesis is entailed by (inferred from) the text.

An appealing approach to such textual inferences is to explicitly transform  $T$  into  $H$ , using a sequence of transformations (Bar-Haim et al., 2007; Harmeling, 2009; Mehdad, 2009; Wang and Manning, 2010; Heilman and Smith, 2010; Stern and Dagan, 2011). Examples of such possible transformations are lexical substitutions (e.g. “letter”  $\rightarrow$  “message”) and predicate-template substitutions (e.g. “X [verb-active] Y”  $\rightarrow$  “Y [verb-passive] by X”), which are based on available knowledge resources. Another example is coreference substitutions, such as replacing “he” with “the employee” if a coreference resolver has detected that these two expressions corefer. Table 1 exemplifies this approach for a particular  $T$ - $H$  pair. The rationale behind this approach is that each transformation step should preserve inference validity, such that each text generated along this process is indeed inferred from the preceding one.

An inherent aspect in transformation-based inference is modeling the certainty that each inference step is valid. This is usually achieved by a cost-based or probabilistic model, which quantifies confidence in the validity of each individual transformation and consequently of the complete chain of inference.

Given a set of possible transformations, there may be many transformation sequences that would transform  $T$  to  $H$ . This creates a very large search space, where systems have to find the “best” transformation sequence – the one of lowest cost, or of highest probability. To the best of our knowledge, this search challenge has not been investigated yet in a substan-

#	Operation	Generated text
0	-	He received the letter from the secretary.
1	Coreference substitution	The employee received the letter from the secretary.
2	X received Y from Z $\rightarrow$ Y was sent to X by Z	The letter was sent to the employee by the secretary.
3	Y [verb-passive] by X $\rightarrow$ X [verb-active] Y	The secretary sent the letter to the employee.
4	X send Y $\rightarrow$ X deliver Y	The secretary delivered the letter to the employee.
5	letter $\rightarrow$ message	The secretary delivered the message to the employee.

Table 1: A sequence of transformations that transform the text “He received the letter from the secretary.” into the hypothesis “The secretary delivered the message to the employee.”. The knowledge required for such transformations is often obtained from available knowledge resources and NLP tools.

tial manner: each of the above-cited works described the search method they used, but none of them tried alternative methods while evaluating search performance. Furthermore, while experimenting with our own open-source inference system, BIUTEE<sup>1</sup>, we observed that search efficiency is a major issue, often yielding practically unsatisfactory run-times.

This paper investigates the search problem in transformation-based textual inference, naturally falling within the framework of heuristic AI (Artificial Intelligence) search. To facilitate such investigation, we formulate a generic search scheme which incorporates many search variants as special cases and enable a meaningful comparison between the algorithms. Under this framework, we identify special characteristics of the textual inference search space, that lead to the development of two novel algorithmic components: a special lookahead method for node expansion, named *local lookahead*, and a gradient-based evaluation function. Together, they yield a new search algorithm, which achieved substantially superior search performance in our evaluations.

The remainder of this paper is organized as follows. Section 2 provides an overview of transformation-based inference systems, AI search algorithms, and search methods realized in prior inference systems. Section 3 formulates the generic search scheme that we have investigated, which covers a broad range of known algorithms, and presents our own algorithmic contributions. These new algorithmic contributions were implemented in our system, BIUTEE. In Section 4 we evaluate them empirically, and show that they improve search efficiency as well as solution’s quality. Search performance is evaluated on two recent RTE benchmarks, in terms

<sup>1</sup>[www.cs.biu.ac.il/~nlp/downloads/biutee](http://www.cs.biu.ac.il/~nlp/downloads/biutee)

of runtime, ability to find lower-cost transformation chains and impact on overall inference.

## 2 Background

Applying sequences of transformations to recognize textual inference was suggested by several works. Such a sequence may be referred to as a *proof*, in the sense that it is used to “prove” the hypothesis from the text. Although various works along this line differ from each other in several respects, many of them share the common challenge of *finding* an optimal proof. The following paragraphs review the major research approaches in this direction. We focus on methods that perform transformations over parse trees, and highlight the search challenge with which they are faced.

### 2.1 Transformation-based textual inference

Several researchers suggested using various types of transformations in order to derive  $H$  from  $T$ . Some suggested a set of predefined transformations, for example, *insertion*, *deletion* and *substitution* of parse-tree nodes, by which any tree can be transformed to any other tree. These transformations were used by the open-source system *EDITS* (Mehdad, 2009), and by (Wang and Manning, 2010). Since the above mentioned transformations are limited in capturing certain interesting and prevalent semantic phenomena, an extended set of tree edit operations (e.g., relabel-edge, move-sibling, etc.) was proposed by Heilman and Smith (2010). Similarly, Harmeling (2009) suggested a heuristic set of 28 transformations, which include various types of node-substitutions as well as restructuring of the entire parse-tree.

In contrast to such predefined sets of transformations, knowledge oriented approaches were sug-



gested by Bar-Haim et al. (2007) and de Salvo Braz et al. (2005). Their transformations are defined by knowledge resources that contain a large amount of *entailment rules*, or *rewrite rules*, which are pairs of parse-tree fragments that entail one another. Typical examples for knowledge resources of such rules are *DIRT* (Lin and Pantel, 2001), and *TEASE* (Szpektor et al., 2004), as well as syntactic transformations constructed manually. In addition, they used knowledge-based lexical substitutions.

However, when only knowledge-based transformations are allowed, transforming the text into the hypothesis is impossible in many cases. This limitation is dealt by our open-source integrated framework, BIUTEE (Stern and Dagan, 2011), which incorporates knowledge-based transformations (entailment rules) with a set of predefined tree-edits. Motivated by the richer structure and search space provided by BIUTEE, we adopted it for our empirical investigations.

The semantic validity of transformation-based inference is usually modeled by defining a *cost* or a *probability estimation* for each transformation. Costs may be defined manually (Kouylekov and Magnini, 2005), but are usually learned automatically (Harmeling, 2009; Mehdad, 2009; Wang and Manning, 2010; Heilman and Smith, 2010; Stern and Dagan, 2011). A global cost (or probability estimation) for a complete sequence of transformations is typically defined as the sum of the costs of the involved transformations.

Finding the lowest cost proof, as needed for determining inference validity, is the focus of our research. Textual inference systems limited to the standard tree-edit operations (insertion, deletion, substitution) can use an exact algorithm that finds the optimal solution in polynomial time under certain constraints (Bille, 2005). Nevertheless, for the extended set of transformations it is unlikely that efficient exact algorithms for finding lowest-cost sequences are available (Heilman and Smith, 2010).

In this harder case, the problem can be viewed as an AI search problem. Each state in the search space is a parse-tree, where the *initial state* is the text parse-tree, the *goal state* is the hypothesis parse-tree, and we search for the shortest (in terms of costs) *path* of transformations from the initial state to the goal state. Next we briefly review major concepts

from the field of AI search and summarize some relevant proposed solutions.

## 2.2 Search Algorithms

Search algorithms find a path from an initial state to a goal state by *expanding* and *generating* states in a search space. The term *generating* a state refers to creating a data structure that represents it, while *expanding* a state means generating all its immediate derivations. In our domain, each state is a parse tree, which is expanded by performing all applicable transformations.

*Best-first search* is a common search framework. It maintains an *open list* (denoted hereafter as OPEN) containing all the generated states that have not been expanded yet. States in OPEN are prioritized by an *evaluation function*,  $f(s)$ . A best-first search algorithm iteratively removes the *best* state (according to  $f(s)$ ) from OPEN, and inserts new states being generated by expanding this best state. The evaluation function is usually a linear combination of the shortest path found from the start state to state  $s$ , denoted by  $g(s)$ , and a heuristic function, denoted by  $h(s)$ , which estimates the cost of reaching a goal state from  $s$ .

Many search algorithms can be viewed as special cases or variations of best-first search. The well-known A\* (Hart et al., 1968) algorithm is a best-first search that uses an evaluation function  $f(s) = g(s) + h(s)$ . Weighted A\* (Pohl, 1970) uses an evaluation function  $f(s) = w \cdot g(s) + h(s)$ , where  $w$  is a parameter, while pure heuristic search uses  $f(s) = h(s)$ . *K*-BFS (Felner et al., 2003) expands  $k$  states in each iteration. Beam search (Furcy and Koenig, 2005; Zhou and Hansen, 2005) limits the number of states stored in OPEN, while Greedy search limits OPEN to contain only the single best state generated in the current iteration.

The search algorithm has crucial impact on the quality of proof found by a textual inference system, as well as on its efficiency. Next, we describe search strategies used in prior works for textual inference.

## 2.3 Search in prior inference models

In spite of being a fundamental problem, prior solutions to the search challenge in textual inference were mostly ad-hoc. Furthermore, there was no investigation of alternative search methods, and no

evaluation of search efficiency and quality was reported. For example, in (Harmeling, 2009) the order by which the transformations are performed is predetermined, and in addition many possible derivations are discarded, to prevent exponential explosion. Handling the search problem in (Heilman and Smith, 2010) was by a variant of greedy search, driven by a similarity measure between the current parse-tree and the hypothesis, while ignoring the cost already paid. In addition, several constraints on the search space were implemented. In the earlier version of BIUTEE (Stern and Dagan, 2011)<sup>2</sup>, a version of beam search was incorporated, named hereafter BIUTEE-orig. This algorithm uses the evaluation function  $f(s) = g(s) + w_i \cdot h(s)$ , where in each iteration ( $i$ ) the value of  $w$  is increased, to ensure successful termination of the search. Nevertheless, its efficiency and quality were not investigated.

In this paper we consider several prominent search algorithms and evaluate their quality. The evaluation concentrates on two measures: the runtime required to find a proof, and proof quality (measured by its cost). In addition to evaluating standard search algorithms we propose two novel components specifically designed for proof-based textual inference and evaluate their contribution.

### 3 Search for Textual Inference

In this section we formalize our search problem and specify a unifying search scheme by which we test several search algorithms in a systematic manner. Then we propose two novel algorithmic components specifically designed for our problem. We conclude by presenting our new search algorithm which combines these two ideas.

#### 3.1 Inference and search space formalization

Let  $t$  be a parse tree, and let  $o$  be a transformation. Applying  $o$  on  $t$ , yielding  $t'$ , is denoted by  $t \vdash_o t'$ . If the underlying meaning of  $t'$  can indeed be inferred from the underlying meaning of  $t$ , then we refer to the application of  $o$  as *valid*. Let  $O = (o_1, o_2, \dots, o_n)$  be a sequence of transformations, such that  $t_0 \vdash_{o_1} t_1 \vdash_{o_2} t_2 \dots \vdash_{o_n} t_n$ . We write  $t_0 \vdash_O t_n$ , and say that  $t_n$  can be *proven* from

<sup>2</sup>More details in [www.cs.biu.ac.il/~nlp/downloads/biutee/search\\_ranlp\\_2011.pdf](http://www.cs.biu.ac.il/~nlp/downloads/biutee/search_ranlp_2011.pdf)

$t_0$  by applying the sequence  $O$ . The *proof* might be valid, if all the transformations involved are valid, or invalid otherwise.

An inference system specifies a *cost*,  $C(o)$ , for each transformation  $o$ . In most systems the costs are automatically learned. The interpretation of a high cost is that it is unlikely that applying  $o$  will be valid. The cost of a sequence  $O = (o_1, o_2, \dots, o_n)$  is defined as  $\sum_{i=1}^n C(o_i)$  (or, in some systems,  $\prod_{i=1}^n C(o_i)$ ). Denoting by  $t_T$  and  $t_H$  the text parse tree and the hypothesis parse tree, a proof system has to find a sequence  $O$  with minimal cost such that  $t_T \vdash_O t_H$ . This forms a search problem of finding the lowest-cost proof among all possible proofs.

The search space is defined as follows. A *state*  $s$  is a parse-tree. The *start state* is  $t_T$  and the *goal state* is  $t_H$ . In some systems any state  $s$  in which  $t_H$  is embedded is considered as goal as well.

Given a state  $s$ , let  $\{o^{(1)}, o^{(2)} \dots o^{(m)}\}$  be  $m$  transformations that can be applied on it. *Expanding*  $s$  means generating  $m$  new states,  $s^{(j)}$ ,  $j = 1 \dots m$ , such that  $s \vdash_{o^{(j)}} s^{(j)}$ . The number  $m$  is called *branching factor*. Our empirical observations on BIUTEE showed that its branching factor ranges from 2-3 for some states to about 30 for other states.

#### 3.2 Search Scheme

Our empirical investigation compares a range prominent search algorithms, described in Section 2. To facilitate such investigation, we formulate them in the following unifying scheme (Algorithm 1).

---

#### Algorithm 1 Unified Search Scheme

---

**Parameters:**  $f(\cdot)$ : state evaluation function  
 $expand(\cdot)$ : state generation function  
**Input:**  $k_{expand}$ : # states expanded in each iteration  
 $k_{maintain}$ : # states in OPEN in each iteration  
 $s_{init}$ : initial state

```

1: OPEN  $\leftarrow \{s_{init}\}$ 
2: repeat
3:   BEST  $\leftarrow k_{expand}$  best (according to  $f$ ) states in OPEN
4:   GENERATED  $\leftarrow \bigcup_{s \in \text{BEST}} expand(s)$ 
5:   OPEN  $\leftarrow (\text{OPEN} \setminus \text{Best}) \cup \text{GENERATED}$ 
6:   OPEN  $\leftarrow k_{maintain}$  best (according to  $f$ ) states in OPEN
7: until BEST contains the goal state

```

---

Initially, the open list, OPEN contains the initial state. Then, the *best*  $k_{expand}$  states from OPEN are chosen, according to the evaluation function  $f(s)$

Algorithm	$f()$	expand()	$k_{\text{maintain}}$	$k_{\text{expand}}$
A*	$g + h$	regular	$\infty$	1
Weighted A*	$g + w \cdot h$	regular	$\infty$	1
K-Weighted A*	$g + w \cdot h$	regular	$\infty$	$k > 1$
Pure Heuristic	$h$	regular	$\infty$	1
Greedy	$g + w \cdot h$	regular	1	1
Beam	$g + h$	regular	$k > 1$	$k > 1$
BIUTEE-orig	$g + w_i \cdot h$	regular	$k > 1$	$k > 1$
LLGS	$\frac{\Delta g}{\Delta h}$	local-lookahead	1	1

Table 2: Search algorithm mapped to the unified search scheme. “Regular” means generating all the states which can be generated by applying a single transformation. Alternative greedy implementations use  $f = h$ .

(line 3), and expanded using the *expansion function*  $\text{expand}(s)$ . In classical search algorithms,  $\text{expand}(s)$  means generating a set of states by applying all the possible state transition operators to  $s$ . Next, we remove from OPEN the states which were expanded, and add the newly generated states. Finally, we keep in OPEN only the best  $k_{\text{maintain}}$  states, according to the evaluation function  $f(s)$  (line 6). This process repeats until the goal state is found in BEST (line 7). Table 2 specifies how known search algorithms, described in Section 2, fit into the unified search scheme.

Since runtime efficiency is crucial in our domain, we focused on improving one of the simple but fast algorithms, namely, greedy search. To improve the quality of the proof found by greedy search, we introduce new algorithmic components for the expansion and evaluation functions, as described in the next two subsections, while maintaining efficiency by keeping  $k_{\text{maintain}} = k_{\text{expand}} = 1$ .

### 3.3 Evaluation function

In most domains, the heuristic function  $h(s)$  estimates the cost of the minimal-cost path from a current state,  $s$ , to a goal state. Having such a function, the value  $g(s) + h(s)$  estimates the expected total cost of a search path containing  $s$ . In our domain, it is yet unclear how to calculate such a heuristic function. Given a state  $s$ , systems typically estimate the difference (the gap) between  $s$  and the hypothesis  $t_H$  (the goal state). In BIUTEE this is quantified by the number of parse-tree nodes and edges of  $t_H$  that do not exist in  $s$ . However, this does not give an

estimation for the expected cost of the path (the sequence of transformations) from  $s$  to the goal state. This is because the number of nodes and edges that can be changed by a single transformation can vary from a single node to several nodes (e.g., by a lexical syntactic entailment rule). Moreover, even if two transformations change the same number of nodes and edges, their costs might be significantly different. Consequently, the measurement of the cost accumulated so far ( $g(s)$ ) and the remaining gap to  $t_H$  ( $h(s)$ ) are unrelated. We note that a more sophisticated heuristic function was suggested by Heilman and Smith (2010), based on tree-kernels. Nevertheless, this heuristic function, serving as  $h(s)$ , is still unrelated to the transformation costs ( $g(s)$ ).

We therefore propose a novel gradient-style function to overcome this difficulty. Our function is designed for a greedy search in which OPEN always contains a single state,  $s$ . Let  $s^j$  be a state generated from  $s$ , the cost of deriving  $s^j$  from  $s$  is  $\Delta_g(s^j) \equiv g(s^j) - g(s)$ . Similarly, the reduction in the value of the heuristic function is defined  $\Delta_h(s^j) \equiv h(s) - h(s^j)$ . Now, we define  $f_\Delta(s^j) \equiv \frac{\Delta_g(s^j)}{\Delta_h(s^j)}$ . Informally, this function measures how costly it is to derive  $s^j$  relative to the obtained decrease in the remaining gap to the goal state. For the edge case in which  $h(s) - h(s^j) \leq 0$ , we define  $f_\Delta(s^j) = \infty$ . Empirically, we show in our experiments that the function  $f_\Delta(s)$  performs better than the traditional functions  $f(s) = g(s) + h(s)$  and  $f_w(s) = g(s) + w \cdot h(s)$  in our domain.

### 3.4 Node expansion method

When examining the proofs produced by the above mentioned algorithms, we observed that in many cases a human could construct proofs that exhibit some internal structure, but were not revealed by the algorithms. Observe, for example, the proof in Table 1. It can be seen that transformations 2,3 and 4 strongly depend on each other. Applying transformation 3 requires first applying transformation 2, and similarly 4 could not be applied unless 2 and 3 are first applied. Moreover, there is no gain in applying transformations 2 and 3, unless transformation 4 is applied as well. On the other hand, transformation 1 does not depend on any other transformation. It may be performed at any point along the proof, and

moreover, changing all other transformations would not affect it.

Carefully examining many examples, we generalized this phenomenon as follows. Often, a sequence of transformations can be decomposed into a set of *coherent subsequences* of transformations, where in each subsequence the transformations strongly depend on each other, while different subsequences are independent. This phenomenon can be utilized in the following way: instead of searching for a complete sequence of transformations that transform  $t_T$  into  $t_H$ , we can iteratively search for independent coherent subsequences of transformations, such that a combination of these subsequences will transform  $t_T$  into  $t_H$ . This is somewhat similar to the technique of applying *macro* operators, which is used in automated planning (Botea et al., 2005) and puzzle solving (Korf, 1985).

One technique for finding such subsequences is to perform, for each state being expanded, a brute-force depth-limited search, also known as *lookahead* (Russell and Norvig, 2010; Bulitko and Lustrek, 2006; Korf, 1990; Stern et al., 2010). However, performing such lookahead might be slow if the branching factor is large. Fortunately, in our domain, coherent subsequences have the following characteristic which can be leveraged: typically, a transformation depends on a previous one only if it is performed over some nodes which were affected by the previous transformation. Accordingly, our proposed algorithm searches for coherent subsequences, in which each subsequent transformation must be applied to nodes that were affected by the previous transformation.

Formally, let  $o$  be a transformation that has been applied on a tree  $t$ , yielding  $t'$ .  $\sigma_{\text{affected}}(o, t')$  denotes the subset of nodes in  $t'$  which were affected (modified or created) by the application of  $o$ .

Next, for a transformation  $o$ , applied on a parse tree  $t$ , we define  $\sigma_{\text{required}}(t, o)$  as the subset of  $t$ 's nodes required for applying  $o$  (i.e., in the absence of these nodes,  $o$  could not be applied).

Finally, let  $t$  be a parse-tree and  $\sigma$  be a subset of its nodes.  $\text{enabled\_ops}(t, \sigma)$  is a function that returns the set of the transformations that can be applied on  $t$ , which require at least one of the nodes in  $\sigma$ . Formally,  $\text{enabled\_ops}(t, \sigma) \equiv \{o \in \mathcal{O} : \sigma \cap \sigma_{\text{required}}(t, o) \neq \emptyset\}$ , where  $\mathcal{O}$  is the set of trans-

formations that can be applied on  $t$ . In our algorithm,  $\sigma$  is the set of nodes that were affected by the preceding transformation of the constructed subsequence.

The recursive procedure described in Algorithm 2 generates all coherent subsequences of lengths up to  $d$ . It should be initially invoked with  $t$  - the current state (parse tree) being expanded,  $\sigma$  - the set of all its nodes,  $d$  - the maximal required length, and  $\emptyset$  as an empty initial sequence. We use  $O \cdot o$  as concatenation of an operation  $o$  to a subsequence  $O$ .

---

#### Algorithm 2 local-lookahead ( $t, \sigma, d, O$ )

---

```

1: if  $d = 0$  then
2:   return  $\emptyset$  (empty-set)
3: end if
4: SUBSEQUENCES  $\leftarrow \emptyset$ 
5: for all  $o \in \text{enabled\_ops}(t, \sigma)$  do
6:   Let  $t \vdash_o t'$ 
7:   Add  $\{O \cdot o\} \cup \text{local-lookahead}(t', \sigma_{\text{affected}}(o, t'), d-1, O \cdot o)$  to SUBSEQUENCES
8: end for
9: return SUBSEQUENCES

```

---

The loop in lines 5 - 8 iterates over transformations that can be applied on the input tree,  $t$ , requiring the same nodes that were affected by the previous transformation of the subsequence being constructed. Note that in the first call  $\text{enabled\_ops}(t, \sigma)$  contain all operations that can be applied on  $t$ , with no restriction. Applying an operation  $o$  results in a new subsequence  $O \cdot o$ . This subsequence will be part of the set of subsequences found by the procedure. In addition, it will be used in the next recursive call as the prefix of additional (longer) subsequences.

### 3.5 Local-lookahead gradient search

We are now ready to define our new algorithm LOCAL-LOOKAHEAD GRADIENT SEARCH (LLGS). In LLGS, like in greedy search,  $k_{\text{maintain}} = k_{\text{expand}} = 1$ .  $\text{expand}(s)$  is defined to return all states generated by subsequences found by the *local-lookahead* procedure, while the evaluation function is defined as  $f = f_{\Delta}$  (see last row of Table 2).

## 4 Evaluation

In this section we first evaluate the search performance in terms of efficiency (run time), the quality

of the found proofs (as measured by proof cost), and overall inference performance achieved through various search algorithms. Finally we analyze the contribution of our two novel components.

#### 4.1 Evaluation settings

We performed our experiments on the last two published RTE datasets: RTE-5 (2009) and RTE-6 (2010). The RTE-5 dataset is composed of a training and test corpora, each containing 600 text-hypothesis pairs, where in half of them the text entails the hypothesis and in the other half it does not. In RTE-6, each of the training and test corpora consists of 10 topics, where each topic contains 10 documents. Each corpus contains a set of hypotheses (211 in the training dataset, and 243 in the test dataset), along with a set of candidate entailing sentences for each hypothesis. The system has to find for each hypothesis which candidate sentences entail it. To improve speed and results, we used the filtering mechanism suggested by (Mirkin et al., 2009), which filters the candidate sentences by the Lucene IR engine<sup>3</sup>. Thus, only top 20 candidates per hypothesis were tested

Evaluation of each of the algorithms was performed by running BIUTEE while replacing BIUTEE-orig with this algorithm. We employed a comprehensive set of knowledge resources (available in BIUTEE’s web site): WordNet (Fellbaum, 1998), Directional similarity (Kotlerman et al., 2010), DIRT (Lin and Pantel, 2001) and generic syntactic rules. In addition, we used coreference substitutions, detected by ArkRef<sup>4</sup>.

We evaluated several known algorithms, described in Table 2 above, as well as BIUTEE-orig. The latter is a strong baseline, which outperforms known search algorithms in generating low cost proofs. We compared all the above mentioned algorithms to our novel one, LLGS.

We used the training dataset for parameter tuning, which controls the trade-off between speed and quality. For weighted A\*, as well as for greedy search, we used  $w = 6.0$ , since, for a few instances, lower values of  $w$  resulted in prohibitive runtime. For beam search we used  $k = 150$ , since higher val-

<sup>3</sup><http://lucene.apache.org>

<sup>4</sup>[www.ark.cs.cmu.edu/ARKref/](http://www.ark.cs.cmu.edu/ARKref/) See (Haghighi and Klein, 2009)

ues of  $k$  did not improve the proof cost on the training dataset. The value of  $d$  in LLGS was set to 3.  $d = 4$  yielded the same proof costs, but was about 3 times slower.

Since lower values of  $w$  could be used by weighted A\* for most instances, we also ran experiments where we varied the value of  $w$  according to the dovetailing method suggested in (Valenzano et al., 2010) (denoted *dovetailing WA\**) as follows. When weighted A\* has found a solution, we reran it with a new value of  $w$ , set to half of the previous value. The idea is to guide the search for lower cost solutions. This process was halted when the total number of states generated by all weighted A\* instances exceeded a predefined constant (set to 10,000).

#### 4.2 Search performance

This experiment evaluates the search algorithms in both efficiency (run-time) and proof quality. Efficiency is measured by the average CPU (Intel Xeon 2.5 GHz) run-time (in seconds) for finding a complete proof for a text-hypothesis instance, and by the average number of generated states along the search. Proof quality is measured by its cost.

The comparison of costs requires that all experiments are performed on the same model which was learned during training. Thus, in the training phase we used the original search of BIUTEE, and then ran the test phase with each algorithm separately. The results, presented in Table 3, show that our novel algorithm, LLGS, outperforms all other algorithms in finding lower cost proofs. The second best is BIUTEE-orig which is much slower by a factor of 3 (on RTE-5) to 8 (on RTE-6)<sup>5</sup>. While inherently fast algorithms, particularly greedy and pure heuristic, achieve faster running times, they achieve lower proof quality, as well as lower overall inference performance (see next subsection).

#### 4.3 Overall inference performance

In this experiment we test whether, and how much, finding better proofs, by a better search algorithm, improves overall success rate of the RTE system. Table 4 summarizes the results (accuracy in RTE-5

<sup>5</sup>Calculating T-test, we found that runtime improvement is statistically significant with  $p < 0.01$ , and  $p < 0.052$  for cost improvement over BIUTEE-orig.

Algorithm	Avg. time	Avg. generated	Avg. cost
Weighted A*	0.22 / <b>0.09</b>	301 / 143	1.11 / 10.52
Dovetailing WA*	7.85 / 8.53	9797 / 9979	1.05 / 10.28
Greedy	0.20 / 0.10	468 / <b>158</b>	1.10 / 10.55
Pure heuristic	<b>0.09</b> / 0.10	<b>123</b> / 167	1.35 / 12.51
Beam search	20.53 / 9.48	43925 / 18992	1.08 / 10.52
BIUTEE-orig	7.86 / 14.61	14749 / 22795	1.03 / 10.28
LLGS	2.76 / 1.72	1722 / 842	<b>0.95</b> / <b>10.14</b>

Table 3: Comparison of algorithms on RTE-5 / RTE-6

and F1 in RTE-6). We see that in RTE-5 LLGS outperforms all other algorithms, and BIUTEE-orig is the second best. This result is statistically significant with  $p < 0.02$  according to *McNemar* test. In RTE-6 we see that although LLGS tends to find lower cost proofs, as shown in Table 3, BIUTEE obtains slightly lower results when utilizing this algorithm.

Algorithm	RTE-5 accuracy %	RTE-6 F1 %
Weighted A*	59.50	48.20
Dovetailing WA*	60.83	49.01
Greedy	60.50	48.56
Pure heuristic	60.83	45.70
Beam search	61.33	48.58
BIUTEE-orig	60.67	<b>49.25</b>
LLGS	<b>64.00</b>	49.09

Table 4: Impact of algorithms on system success rate

#### 4.4 Component evaluation

In this experiment we examine separately our two novel components. We examined  $f_{\Delta}$  by running LLGS with alternative evaluation functions. The results, displayed in Table 5, show that using  $f_{\Delta}$  yields better proofs and also improves run time.

$f$	Avg. time	Avg. cost	Accuracy %
$f = g + h$	3.28	1.06	61.50
$f = g + w \cdot h$	3.30	1.07	61.33
$f = f_{\Delta}$	<b>2.76</b>	<b>0.95</b>	<b>64.0</b>

Table 5: Impact of  $f_{\Delta}$  on RTE-5.  $w = 6.0$ . Accuracy obtained by retraining with corresponding  $f$ .

Our *local-lookahead* (Subsection 3.4) was examined by running LLGS with alternative node expansion methods. One alternative to local-lookahead is standard expansion by generating all immediate derivations. Another alternative is to use the standard lookahead, in which a brute-force depth-limited

search is performed in each iteration, termed here “exhaustive lookahead”. The results, presented in Table 6, show that by avoiding any type of lookahead one can achieve fast runtime, while compromising proof quality. On the other hand, both exhaustive and local lookahead yield better proofs and accuracy, while local lookahead is more than 4 times faster than exhaustive lookahead.

lookahead	Avg. time	Avg. cost	Accuracy (%)
exhaustive	13.22	<b>0.95</b>	<b>64.0</b>
local	2.76	<b>0.95</b>	<b>64.0</b>
none	<b>0.24</b>	0.97	62.0

Table 6: Impact of local and global lookahead on RTE-5. Accuracy obtained by retraining with the corresponding lookahead method.

## 5 Conclusion

In this paper we investigated the efficiency and proof quality obtained by various search algorithms. Consequently, we observed special phenomena of the search space in textual inference and proposed two novel components yielding a new search algorithm, targeted for our domain. We have shown empirically that (1) this algorithm improves run time by factors of 3-8 relative to BIUTEE-orig, and by similar factors relative to standard AI-search algorithms that achieve similar proof quality; and (2) outperforms all other algorithms in finding low cost proofs.

In future work we plan to investigate other search paradigms, e.g., Monte-Carlo style approaches (Kocsis and Szepesvári, 2006), which do not fall under the AI search scheme covered in this paper. In addition, while our novel components were motivated by the search space of textual inference, we foresee their potential utility in other application areas for search, such as automated planning and scheduling.

## Acknowledgments

This work was partially supported by the Israel Science Foundation grant 1112/08, the PASCAL-2 Network of Excellence of the European Community FP7-ICT-2007-1-216886, and the European Community’s Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT).

## References

- Roy Bar-Haim, Ido Dagan, Iddo Greental, and Eyal Shnarch. 2007. Semantic inference at the lexical-syntactic level. In *Proceedings of AAAI*.
- Philip Bille. 2005. A survey on tree edit distance and related problems. *Theoretical Computer Science*.
- Adi Botea, Markus Enzenberger, Martin Müller, and Jonathan Schaeffer. 2005. Macro-FF: Improving ai planning with automatically learned macro-operators. *J. Artif. Intell. Res. (JAIR)*, 24:581–621.
- Vadim Bulitko and Mitja Lustrek. 2006. Lookahead pathology in real-time path-finding. In *proceedings of AAAI*.
- Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of MLCW*.
- Rodrigo de Salvo Braz, Roxana Girju, Vasin Punyakanok, Dan Roth, and Mark Sammons. 2005. An inference model for semantic entailment in natural language. In *Proceedings of AAAI*.
- Christiane Fellbaum, editor. 1998. *WordNet An Electronic Lexical Database*. The MIT Press, May.
- Ariel Felner, Sarit Kraus, and Richard E. Korf. 2003. KBFS: K-best-first search. *Ann. Math. Artif. Intell.*, 39(1-2):19–39.
- David Furcy and Sven Koenig. 2005. Limited discrepancy beam search. In *proceedings of IJCAI*.
- Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of EMNLP*.
- Stefan Harmeling. 2009. Inferring textual entailment with a probabilistically sound calculus. *Natural Language Engineering*.
- Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, SSC-4(2):100–107.
- Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Proceedings of NAACL*.
- Levente Kocsis and Csaba Szepesvári. 2006. Bandit based monte-carlo planning. In *proceedings of ECML*.
- Richard E. Korf. 1985. Macro-operators: A weak method for learning. *Artif. Intell.*, 26(1):35–77.
- Richard E. Korf. 1990. Real-time heuristic search. *Artif. Intell.*, 42(2-3):189–211.
- Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*.
- Milen Kouylekov and Bernardo Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *Proceedings of Pascal Challenges Workshop on Recognising Textual Entailment*.
- Dekang Lin and Patrick Pantel. 2001. DIRT - discovery of inference rules from text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.
- Yashar Mehdad. 2009. Automatic cost estimation for tree edit distance using particle swarm optimization. In *Proceedings of the ACL-IJCNLP*.
- Shachar Mirkin, Roy Bar-Haim, Jonathan Berant, Ido Dagan, Eyal Shnarch, Asher Stern, and Idan Szpektor. 2009. Addressing discourse and document structure in the rte search task. In *Proceedings of TAC*.
- Ira Pohl. 1970. Heuristic search viewed as path finding in a graph. *Artificial Intelligence*, 1(3-4):193 – 204.
- Stuart Russell and Peter Norvig. 2010. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 3rd edition.
- Asher Stern and Ido Dagan. 2011. A confidence model for syntactically-motivated entailment proofs. In *Proceedings of RANLP*.
- Roni Stern, Tamar Kulberis, Ariel Felner, and Robert Holte. 2010. Using lookaheads with optimal best-first search. In *proceedings of AAAI*.
- Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP*.
- Richard Anthony Valenzano, Nathan R. Sturtevant, Jonathan Schaeffer, Karen Buro, and Akihiro Kishimoto. 2010. Simultaneously searching with multiple settings: An alternative to parameter tuning for suboptimal single-agent search algorithms. In *proceedings of ICAPS*.
- Mengqiu Wang and Christopher D. Manning. 2010. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of COLING*.
- Rong Zhou and Eric A. Hansen. 2005. Beam-stack search: Integrating backtracking with beam search. In *proceedings of ICAPS*.

# Maximum Expected BLEU Training of Phrase and Lexicon Translation Models

**Xiaodong He**

Microsoft Research  
One Microsoft Way, Redmond, WA, USA  
xiaohe@microsoft.com

**Li Deng**

Microsoft Research  
One Microsoft Way, Redmond, WA, USA  
deng@microsoft.com

## Abstract

This paper proposes a new discriminative training method in constructing phrase and lexicon translation models. In order to reliably learn a myriad of parameters in these models, we propose an expected BLEU score-based utility function with KL regularization as the objective, and train the models on a large parallel dataset. For training, we derive growth transformations for phrase and lexicon translation probabilities to iteratively improve the objective. The proposed method, evaluated on the Europarl German-to-English dataset, leads to a 1.1 BLEU point improvement over a state-of-the-art baseline translation system. In IWSLT 2011 Benchmark, our system using the proposed method achieves the best Chinese-to-English translation result on the task of translating TED talks.

## 1. Introduction

Discriminative training is an active area in statistical machine translation (SMT) (e.g., Och et al., 2002, 2003, Liang et al., 2006, Blunsom et al., 2008, Chiang et al., 2009, Foster et al., 2010, Xiao et al., 2011). Och (2003) proposed using a log-linear model to incorporate multiple features for translation, and proposed a minimum error rate training (MERT) method to train the feature weights to optimize a desirable translation metric.

While the log-linear model itself is discriminative, the phrase and lexicon translation features, which are among the most important components of SMT, are derived from either generative models or heuristics (Koehn et al., 2003, Brown et al., 1993). Moreover, the

parameters in the phrase and lexicon translation models are estimated by relative frequency or maximizing joint likelihood, which may not correspond closely to the translation measure, e.g., bilingual evaluation understudy (BLEU) (Papineni et al., 2002). Therefore, it is desirable to train all these parameters to directly maximize an objective that directly links to translation quality.

However, there are a large number of parameters in these models, making discriminative training for them non-trivial (e.g., Liang et al., 2006, Chiang et al., 2009). Liang et al. (2006) proposed a large set of lexical and Part-of-Speech features and trained the model weights associated with these features using perceptron. Since many of the reference translations are non-reachable, an empirical *local updating* strategy had to be devised to fix this problem by picking a *pseudo* reference. Many such non-desirable heuristics led to moderate gains reported in that work. Chiang et al. (2009) improved a syntactic SMT system by adding as many as ten thousand syntactic features, and used Margin Infused Relaxed Algorithm (MIRA) to train the feature weights. However, the number of parameters in common phrase and lexicon translation models is much larger.

In this work, we present a new, highly effective discriminative learning method for phrase and lexicon translation models. The training objective is an expected BLEU score, which is closely linked to translation quality. Further, we apply a Kullback–Leibler (KL) divergence regularization to prevent over-fitting.

For effective optimization, we derive updating formulas of growth transformation (GT) for phrase and lexicon translation probabilities. A GT is a transformation of the probabilities that guarantees strict non-decrease of the objective over each GT iteration unless a local maximum is reached. A



similar GT technique has been successfully used in speech recognition (Gopalakrishnan et al., 1991, Povey, 2004, He et al., 2008). Our work demonstrates that it works with large scale discriminative training of SMT model as well.

Our work is based on a phrase-based SMT system. Experiments on the Europarl German-to-English dataset show that the proposed method leads to a 1.1 BLEU point improvement over a strong baseline. The proposed method is also successfully evaluated on the IWSLT 2011 benchmark test set, where the task is to translate TED talks ([www.ted.com](http://www.ted.com)). Our experimental results on this open-domain spoken language translation task show that the proposed method leads to significant translation performance improvement over a state-of-the-art baseline, and the system using the proposed method achieved the best single system translation result in the Chinese-to-English MT track.

## 2. Related Work

One best known approach in discriminative training for SMT is proposed by Och (2003). In that work, multiple features, most of them are derived from generative models, are incorporated into a log-linear model, and the relative weights of them are tuned discriminatively on a small tuning set. However, in practice, this approach only works with a handful of parameters.

More closely related to our work, Liang et al. (2006) proposed a large set of lexical and Part-of-Speech features in addition to the phrase translation model. Weights of these features are trained using perceptron on a training set of 67K sentences. In that paper, the authors pointed out that forcing the model to update towards the reference translation could be problematic. This is because the hidden structure such as phrase segmentation and alignment could be abused if the system is forced to produce a reference translation. Therefore, instead of pushing the parameter update towards the reference translation (a.k.a. *bold updating*), the author proposed a *local updating* strategy where the model parameters are updated towards a pseudo-reference (i.e., the hypothesis in the n-best list that gives the best BLEU score). Experimental results showed that their approach outperformed a baseline by 0.8 BLEU point when using monotonic decoding, but there was no

significant gain over a stronger baseline with a full-distortion model. In our work, we use the expectation of BLEU scores as the objective. This avoids the heuristics of picking the updating reference and therefore gives a more principal way of setting the training objective.

As another closely related study, Chiang et al. (2009) incorporated about ten thousand syntactic features in addition to the baseline features. The feature weights are trained on a tuning set with 2010 sentences using MIRA. In our work, we have many more parameters to train, and the training is conducted on the entire training corpora. Our GT based optimization algorithm is highly parallelizable and efficient, which is the key for large scale discriminative training.

As a further related work, Rosti et al. (2011) have proposed using differentiable expected BLEU score as the objective to train system combination parameters. Other work related to the computation of expected BLEU in common with ours includes minimum Bayes risk approaches (Smith and Eisner 2006, Tromble et al., 2008) and lattice-based MERT (Macherey et al., 2008). In these earlier work, however, the phrase and lexicon translation models used remained unchanged.

Another line of research that is closely related to our work is phrase table refinement and pruning. Wuebker et al. (2010) proposed a method to train the phrase translation model using Expectation-Maximization algorithm with a *leave-one-out* strategy. The parallel sentences were forced to be aligned at the phrase level using the phrase table and other features as in a decoding process. Then the phrase translation probabilities were estimated based on the phrase alignments. To prevent overfitting, the statistics of phrase pairs from a particular sentence was excluded from the phrase table when aligning that sentence. However, as pointed out by Liang et al (2006), the same problem as in the *bold updating* existed, i.e., forced alignment between a source sentence and its reference translation was tricky, and the proposed alignment was likely to be unreliable. The method presented in this paper is free from this problem.

## 3. Phrase-based Translation System

The translation process of phrase-based SMT can be briefly described in three steps: segment source sentence into a sequence of phrases, translate each

source phrase to a target phrase, re-order target phrases into target sentence (Koehn et al., 2003).

In decoding, the optimal translation  $\hat{E}$  given the source sentence  $F$  is obtained according to

$$\hat{E} = \operatorname{argmax}_E P(E|F) \quad (1)$$

where

$$P(E|F) = \frac{1}{Z} \exp \left\{ \sum_m \lambda_m \log h_m(E, F) \right\} \quad (2)$$

and  $Z = \sum_E \exp \{ \sum_m \lambda_m \log h_m(E, F) \}$  is the normalization denominator to ensure that the probabilities sum to one. Note that we define the feature functions  $\{h_m(E, F)\}$  in log domain to simplify the notation in later sections. Feature weights  $\lambda = \{\lambda_m\}$  are usually tuned by MERT.

Features used in a phrase-based system usually include LM, reordering model, word and phrase counts, and phrase and lexicon translation models. Given the focus of this paper, we review only the phrase and lexicon translation models below.

### 3.1. Phrase translation model

A set of phrase pairs are extracted from word-aligned parallel corpus according to phrase extraction rules (Koehn et al., 2003). Phrase translation probabilities are then computed as relative frequencies of phrases over the training dataset. i.e., the probability of translating a source phrase  $\tilde{f}$  to a target phrase  $\tilde{e}$  is computed by

$$p(\tilde{e}|\tilde{f}) = \frac{C(\tilde{e}, \tilde{f})}{C(\tilde{f})} \quad (3)$$

where  $C(\tilde{e}, \tilde{f})$  is the joint counts of  $\tilde{e}$  and  $\tilde{f}$ , and  $C(\tilde{f})$  is the marginal counts of  $\tilde{f}$ .

In translation, the input sentence is segmented into  $K$  phrases, and the source-to-target forward phrase (FP) translation feature is scored as:

$$h_{FP}(E, F) = \prod_k p(\tilde{e}_k|\tilde{f}_k) \quad (4)$$

where  $\tilde{e}_k$  and  $\tilde{f}_k$  are the  $k$ -th phrase in  $E$  and  $F$ , respectively. The target-to-source (backward) phrase translation model is defined similarly.

### 3.2. Lexicon translation model

There are several variations in lexicon translation features (Ayan and Dorr 2006, Koehn et al., 2003, Quirk et al., 2005). We use the word translation table from IBM Model 1 (Brown et al., 1993) and compute the sum over all possible word alignments within a phrase pair without normalizing for length (Quirk et al., 2005). The source-to-target forward lexicon (FL) translation feature is:

$$h_{FL}(E, F) = \prod_k \prod_m \sum_r p(e_{k,m}|f_{k,r}) \quad (5)$$

where  $e_{k,m}$  is the  $m$ -th word of the  $k$ -th target phrase  $\tilde{e}_k$ ,  $f_{k,r}$  is the  $r$ -th word in the  $k$ -th source phrase  $\tilde{f}_k$ , and  $p(e_{k,m}|f_{k,r})$  is the probability of translating word  $f_{k,r}$  to word  $e_{k,m}$ . In IBM model 1, these probabilities are learned via maximizing a joint likelihood between the source and target sentences. The target-to-source (backward) lexicon translation model is defined similarly.

## 4. Maximum Expected-BLEU Training

### 4.1. Objective function

We denote by  $\theta$  the set of all the parameters to be optimized, including forward phrase and lexicon translation probabilities and their backward counterparts. For simplification of notation,  $\theta$  is formed as a matrix, where its elements  $\{\theta_{ij}\}$  are probabilities subject to  $\sum_j \theta_{ij} = 1$ . E.g., each row is a probability distribution.

The utility function over the entire training set is defined as:

$$U(\theta) = \sum_{E_1, \dots, E_N} P_\theta(E_1, \dots, E_N | F_1, \dots, F_N) \left( \sum_{n=1}^N BLEU(E_n, E_n^*) \right) \quad (6)$$

where  $N$  is the number of sentences in the training set,  $E_n^*$  is the reference translation of the  $n$ -th source sentence  $F_n$ , and  $E_n \in Hyp(F_n)$  that denotes the list of translation hypotheses of  $F_n$ . Since the sentences are independent with each other, the joint posterior can be decomposed:

$$P_\theta(E_1, \dots, E_N | F_1, \dots, F_N) = \prod_{n=1}^N P_\theta(E_n | F_n) \quad (7)$$

and  $P_{\theta}(E_n|F_n)$  is the posterior defined in (2), the subscript  $\theta$  indicates that it is computed based on the parameter set  $\theta$ .  $U(\theta)$  is proportional (with a factor of  $N$ ) to the expected sentence BLEU score over the entire training set, i.e., after some algebra,

$$U(\theta) = \sum_{n=1}^N \sum_{E_n} P_{\theta}(E_n|F_n) BLEU(E_n, E_n^*)$$

In a phrase-based SMT system, the total number of parameters of phrase and lexicon translation models, which we aim to learn discriminatively, is very large (see Table 1). Therefore, regularization is critical to prevent over-fitting. In this work, we regularize the parameters with KL regularization.

KL divergence is commonly used to measure the distance between two probability distributions. For the whole parameter set  $\theta$ , the KL regularization is defined in this work as the sum of KL divergence over the entire parameter space:

$$KL(\theta^0||\theta) = \sum_i \sum_j \theta_{ij}^0 \log \frac{\theta_{ij}^0}{\theta_{ij}} \quad (8)$$

where  $\theta^0$  is a constant prior parameter set. In training, we want to improve the utility function while keeping the changes of the parameters from  $\theta^0$  at minimum. Therefore, we design the objective function to be maximized as:

$$O(\theta) = \log U(\theta) - \tau \cdot KL(\theta^0||\theta) \quad (9)$$

where the prior model  $\theta^0$  in our approach is the relative-frequency-based phrase translation model and the maximum-likelihood-estimated IBM model 1 (word translation model).  $\tau$  is a hyper-parameter controlling the degree of regularization.

## 4.2. Optimization

In this section, we derived GT formulas for iteratively updating the parameters so as to optimize objective (9). GT is based on extended Baum-Welch (EBW) algorithm first proposed by Gopalakrishnan et al. (1991) and commonly used in speech recognition (e.g., He et al. 2008).

### 4.2.1. Extended Baum-Welch Algorithm

Baum-Eagon inequality (Baum and Eagon, 1967) gives the GT formula to iteratively maximize positive-coefficient polynomials of random

variables that are subject to sum-to-one constants. Baum-Welch algorithm is a model update algorithm for hidden Markov model which uses this GT. Gopalakrishnan et al. (1991) extended the algorithm to handle rational function, i.e., a ratio of two polynomials, which is more commonly encountered in discriminative training.

Here we briefly review EBW. Assuming a set of random variables  $\mathbf{p} = \{p_{ij}\}$  that subject to the constraint that  $\sum_j p_{ij} = 1$ , and assume  $g(\mathbf{p})$  and  $h(\mathbf{p})$  are two positive polynomial functions of  $\mathbf{p}$ , a GT of  $\mathbf{p}$  for the rational function  $r(\mathbf{p}) = \frac{g(\mathbf{p})}{h(\mathbf{p})}$  can be obtained through the following two steps:

i) Construct the auxiliary function:

$$f(\mathbf{p}) = g(\mathbf{p}) - r(\mathbf{p}')h(\mathbf{p}) \quad (10)$$

where  $\mathbf{p}'$  are the values from the previous iteration. Increasing  $f$  guarantees an increase of  $r$ , i.e.,  $h(\mathbf{p}) > 0$  and  $r(\mathbf{p}) - r(\mathbf{p}') = \frac{1}{h(\mathbf{p})}(f(\mathbf{p}) - f(\mathbf{p}'))$ .

ii) Derive GT formula for  $f(\mathbf{p})$

$$p_{ij} = \frac{p'_{ij} \left. \frac{\partial f(\mathbf{p})}{\partial p_{ij}} \right|_{\mathbf{p}=\mathbf{p}'} + D \cdot p'_{ij}}{\sum_j p'_{ij} \left. \frac{\partial f(\mathbf{p})}{\partial p_{ij}} \right|_{\mathbf{p}=\mathbf{p}'} + D} \quad (11)$$

where  $D$  is a smoothing factor.

### 4.2.2. GT of Translation Models

Now we derive the GTs of translation models for our objective. Since maximizing  $O(\theta)$  is equivalent to maximizing  $e^{O(\theta)}$ , we have the following auxiliary function:

$$R(\theta) = U(\theta) e^{-\tau \cdot KL(\theta^0||\theta)} \quad (12)$$

After substituting (2) and (7) into (6), and drop optimization irrelevant terms in KL regularization, we have  $R(\theta)$  in a rational function form:

$$R(\theta) = \frac{G(\theta) \cdot J(\theta)}{H(\theta)} \quad (13)$$

where  $H(\theta) = \sum_{E_1, \dots, E_N} \prod_{n=1}^N \prod_m h_m^{\lambda_m}(E_n, F_n)$ ,  $J(\theta) = \prod_i \prod_j \theta_{ij}^{\tau \theta_{ij}^0}$ , and  $G(\theta) =$

$\sum_{E_1, \dots, E_N} \prod_{n=1}^N \prod_m h_m^{\lambda_m}(E_n, F_n)$  ( $\sum_{n=1}^N BLEU(E_n, E_n^*)$ ) are all positive polynomials of  $\theta$ . Therefore, we can follow the two steps of EBW to derive the GT formulas for  $\theta$ .

If we denote by  $p_{ij}$  the probability of translating the source phrase  $i$  to the target phrase  $j$ . Then, the updating formula is (derivation omitted):

$$p_{ij} = \frac{\sum_n \sum_{E_n} \gamma_{FP}(E_n, n, i, j) + U(\theta') \tau_{FP} p_{ij}^0 + D_i p'_{ij}}{\sum_n \sum_{E_n} \sum_j \gamma_{FP}(E_n, n, i, j) + U(\theta') \tau_{FP} + D_i} \quad (14)$$

where  $\tau_{FP} = \tau / \lambda_{FP}$  and  $\gamma_{FP}(E_n, n, i, j) = P_{\theta'}(E_n | F_n) \cdot [BLEU(E_n, E_n^*) - U_n(\theta')] \cdot \sum_k \mathbf{1}(\tilde{f}_{n,k} = i, \tilde{e}_{n,k} = j)$ . In which  $U_n(\theta')$  takes a form similar to (6), but is the expected BLEU score for sentence  $n$  using models from the previous iteration.  $\tilde{f}_{n,k}$  and  $\tilde{e}_{n,k}$  are the  $k$ -th phrases of  $F_n$  and  $E_n$ , respectively.

The smoothing factor set of  $D_i$  according to the Baum-Eagon inequality is usually far too large for practical use. In practice, one general guide of setting  $D_i$  is to make all updated value positive. Similar to (Povey 2004), we set  $D_i$  by

$$D_i = \sum_n \sum_{E_n} \sum_j \max(0, -\gamma_{FP}(E_n, n, i, j)) \quad (15)$$

to ensure the denominator of (15) is positive. Further, we set a low-bound of  $D_i$  as  $\max_j \left\{ \frac{-\sum_n \sum_{E_n} \gamma_{FP}(E_n, n, i, j)}{p'_{ij}} \right\}$  to guarantee the numerator to be positive.

We denote by  $l_{ij}$  the probability of translating the source word  $i$  to the target word  $j$ . Then following the same derivation, we get the updating formula for forward lexicon translation model:

$$l_{ij} = \frac{\sum_n \sum_{E_n} \gamma_{FL}(E_n, n, i, j) + U(\theta') \tau_{FL} l_{ij}^0 + D_i l'_{ij}}{\sum_n \sum_{E_n} \sum_j \gamma_{FL}(E_n, n, i, j) + U(\theta') \tau_{FL} + D_i} \quad (16)$$

where  $\tau_{FL} = \tau / \lambda_{FL}$  and  $\gamma_{FL}(E_n, n, i, j) = P_{\theta'}(E_n | F_n) \cdot [BLEU(E_n, E_n^*) - U_n(\theta')] \cdot \sum_k \sum_m \mathbf{1}(e_{n,k,m} = j) \gamma(n, k, m, i)$ , and  $\gamma(n, k, m, i) = \frac{\sum_r \mathbf{1}(f_{n,k,r} = i) p'(e_{n,k,m} | f_{n,k,r})}{\sum_r p'(e_{n,k,m} | f_{n,k,r})}$ , in which  $f_{n,k,r}$  and  $e_{n,k,m}$  are the  $r$ -th and  $m$ -th word in the  $k$ -th phrase of the source sentence  $F_n$  and the target hypothesis  $E_n$ , respectively. Value of  $D_i$  is set in a

way similar to (15).

GTs for updating backward phrase and lexicon translation models can be derived in a similar way, and is omitted here.

### 4.3. Implementation issues

#### 4.3.1. Normalizing $\lambda$

The posterior  $p_{\theta'}(E_n | F_n)$  in the model updating formula is computed according to (2). In decoding, only the relative values of  $\lambda$  matters. However, the absolute value will affect the posterior distribution, e.g., an overly large absolute value of  $\lambda$  would lead to a very sharp posterior distribution. In order to control the sharpness of the posterior distribution, we normalize  $\lambda$  by its L1 norm:

$$\hat{\lambda}_m = \frac{\lambda_m}{\sum_m |\lambda_m|} \quad (17)$$

#### 4.3.2. Computing the sentence BLEU score

The commonly used BLEU-4 score is computed by

$$BLEU-4 = BP \cdot \exp\left(\frac{1}{4} \sum_{n=1}^4 \log p_n\right) \quad (18)$$

In the updating formula, we need to compute the sentence-level  $BLEU(E_n, E_n^*)$ . Since the matching count may be sparse at the sentence level, we smooth raw precisions of high-order  $n$ -grams by:

$$p_n = \frac{\#(n\text{-gram matched}) + \eta \cdot p_n^0}{\#(n\text{-gram}) + \eta} \quad (19)$$

where  $p_n^0$  is the prior value of  $p_n$ ,  $\eta$  is a smoothing factor usually takes a value of 5 and  $p_n^0$  can be set by  $p_n^0 = p_{n-1} \cdot p_{n-1} / p_{n-2}$ , for  $n = 3, 4$ .  $p_1$  and  $p_2$  are estimated empirically. Brevity penalty (BP) also plays a key role. Instead of clip it at 1, we use a non-clipped BP,  $BP = e^{(1-p_n^2)}$ , for sentence-level BLEU<sup>1</sup>. We further scale the reference length,  $r$ , by a factor such that the total length of references on the training set equals that of the baseline output<sup>2</sup>.

<sup>1</sup> This is to better approximate corpus-level BLEU, i.e., as discussed in (Chiang, et al., 2008), the per-sentence BP might effectively exceed unity in corpus-level BLEU computation.

<sup>2</sup> This is to focus the training on improving BLEU by improving  $n$ -gram match instead of by improving BP, e.g., this makes the BP of the baseline output already being perfect.

### 4.3.3. Training procedure

The parameter set  $\theta$  is optimized on the training set while the feature weights  $\lambda$  are tuned on a small tuning set<sup>3</sup>. Since  $\theta$  and  $\lambda$  affect the training of each other, we train them in alternation. I.e., at each iteration, we first fix  $\lambda$  and update  $\theta$ , then we re-tune  $\lambda$  given the new  $\theta$ . Due to mismatch between training and tuning data, the training process might not always converge. Therefore, we need a validation set to determine the stop point of training. At the end,  $\theta$  and  $\lambda$  that give the best score on the validation set are selected and applied to the test set. Fig. 1 gives a summary of the training procedure. Note that step 2 and 4 are parallelize-able across multiple processors.

- |   |
|---|
| <ol style="list-style-type: none"> <li>1. Build the baseline system, estimate <math>\{\theta, \lambda\}</math>.</li> <li>2. Decode N-best list for training corpus using the baseline system, compute <math>BLEU(E_n, E_n^*)</math>.</li> <li>3. set <math>\theta' = \theta, \lambda' = \lambda</math>.</li> <li>4. Max expected BLEU training             <ol style="list-style-type: none"> <li>a. Go through the training set.                 <ol style="list-style-type: none"> <li>i. Compute <math>P_{\theta'}(E_n F_n)</math> and <math>U_n(\theta')</math>.</li> <li>ii. Accumulate statistics <math>\{\gamma\}</math>.</li> </ol> </li> <li>b. Update: <math>\theta' \rightarrow \theta</math> by one iteration of GT.</li> </ol> </li> <li>5. MERT on the tuning set: <math>\lambda' \rightarrow \lambda</math>.</li> <li>6. Test on the validation set using <math>\{\theta, \lambda\}</math>.</li> <li>7. Go to step 3 unless training converges or reaches a certain number of iterations.</li> <li>8. Pick the best <math>\{\theta, \lambda\}</math> on the validation set.</li> </ol> |
|---|

Figure 1. The max expected-BLEU training algorithm.

## 5. Evaluation

In evaluating the proposed method, we use two separate datasets. We first describe the experiments with the *Europarl* dataset (Koehn 2002), followed by the experiments with the more recent IWSLT-2011 task (Federico et al., 2011).

### 5.1 Experimental setup in the Europarl task

In evaluating the proposed method, we use two separate datasets. First, we conduct experiments on the *Europarl* German-to-English dataset. The training corpus contains 751K sentence pairs, 21 words per sentence on average. 2000 sentences are provided in the development set. We use the first 1000 sentences for  $\lambda$  tuning, and the rest for validation. The test set consists of 2000 sentences.

<sup>3</sup> Usually, the tuning set matches the test condition better, and therefore is preferable for  $\lambda$  tuning.

To build the baseline phrase-based SMT system, we first perform word alignment on the training set using a hidden Markov model with lexicalized distortion (He 2007), then extract the phrase table from the word aligned bilingual texts (Koehn et al., 2003). The maximum phrase length is set to four. Other models used in the baseline system include lexicalized ordering model, word count and phrase count, and a 3-gram LM trained on the English side of the parallel training corpus. Feature weights are tuned by MERT. A fast beam-search phrase-based decoder (Moore and Quirk 2007) is used and the distortion limit is set to four. Details of the phrase and lexicon translation models are given in Table 1. This baseline achieves a BLEU score of 26.22% on the test set. This baseline system is also used to generate a 100-best list of the training corpus during maximum expected BLEU training.

Translation model	# parameters
Phrase models (fore. & back.)	9.2 M
Lexicon model (IBM-1 src-to-tgt)	12.9 M
Lexicon model (IBM-1 tgt-to-src)	11.9 M

Table 1. Summary of phrase and lexicon translation models

### 5.2 Experimental results on the Europarl task

During training, we first tune the regularization factor  $\tau$  based on the performance on the validation set. For simplicity reasons, the tuning of  $\tau$  makes use of only the phrase translation models. Table 2 reports the BLEU scores and gains over the baseline given different values of  $\tau$ . The results highlight the importance of regularization. While  $\tau = 5 \times 10^{-5}$  gives the best score on the validation set, the gain is shown to be substantially reduced to merely 0.2 BLEU point when  $\tau = 0$ , i.e., no regularization. We set the optimal value of  $\tau = 5 \times 10^{-5}$  in all remaining experiments.

Test on Validation Set	BLEU%	$\Delta BLEU\%$
Baseline	26.70	--
$\tau = 0$ (no regularization)	26.91	+0.21
$\tau = 1 \times 10^{-5}$	27.31	+0.61
$\tau = 5 \times 10^{-5}$	27.44	+0.74
$\tau = 10 \times 10^{-5}$	27.27	+0.57

Table 2. Results on degrees of regularizations. BLEU scores are reported on the validation set.  $\Delta BLEU$  denotes the gain over the baseline.

Fixing the optimal regularization factor  $\tau$ , we then study the relationship between the expected

sentence-level BLEU (Exp. BLEU) score of N-best lists and the corpus-level BLEU score of 1-best translations. The conjectured close relationship between the two is important in justifying our use of the former as the training objective. Fig. 2 shows these two scores on the training set over training iterations. Since the expected BLEU is affected by  $\lambda$  strongly, we fix the value of  $\lambda$  in order to make the expected BLEU comparable across different iterations. From Fig. 2 it is clear that the expected BLEU score correlates strongly with the real BLEU score, justifying its use as our training objective.

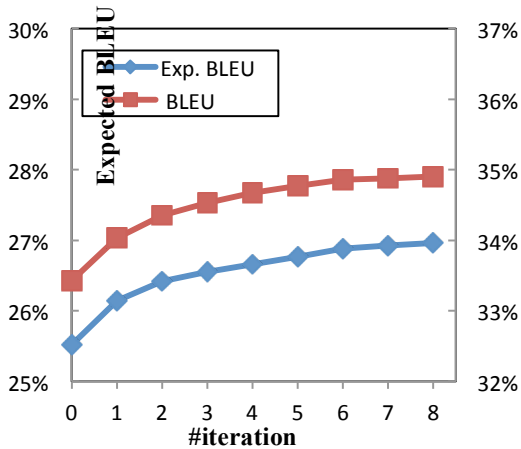


Figure 2. Expected sentence BLEU and 1-best corpus BLEU on the 751K sentence of training data.

Next, we study the effects of training the phrase translation probabilities and the lexicon translation probabilities according to the GT formulas presented in the preceding section. The breakdown results are shown in Table 3. Compared with the baseline, training phrase or lexicon models alone gives a gain of 0.7 and 0.5 BLEU points, respectively, on the test set. For a full training of both phrase and lexicon models, we adopt two learning schedules: update both models together at each iteration (*simultaneously*), or update them in two stages (*two-stage*), where the phrase models are trained first until reaching the best score on the validation set and then the lexicon models are trained. Both learning schedules give significant improvements over the baseline and also over training phrase or lexicon models alone. The *two-stage* training of both models gives the best result of 27.33%, outperforming the baseline by 1.1 BLEU points.

More detail of the two-stage training is provided in Fig. 3, where BLEU scores in each stage are shown as a function of the GT training iteration. The phrase translation probabilities (PT) are trained alone in the first stage, shown in blue color. After five iterations, the BLEU score on the validation set reaches the peak value, with further iteration giving BLEU score fluctuation. Hence, we perform lexicon model (LEX) training starting from the sixth iteration with the corresponding BLEU scores shown in red color in Fig. 3. The BLEU score is further improved by 0.4 points after additional three iterations of training the lexicon models. In total, nine iterations are performed to complete the two-stage GT training of all phrase and lexicon models.

BLEU (%)	validation	test
Baseline	26.70	26.22
Train phrase models alone	27.44	26.94*
Train lexicon models alone	27.36	26.71
Both models: <i>simultaneously</i>	27.65	27.13*
Both models: <i>two-stage</i>	27.82	27.33*

Table 3. Results on the Europarl German-to-English dataset. The BLEU measures from various settings of maximum expected BLEU training are compared with the baseline, where \* denotes that the gain over the baseline is statistically significant with a significance level > 99%, measured by paired bootstrap resampling method proposed by Koehn (2004).

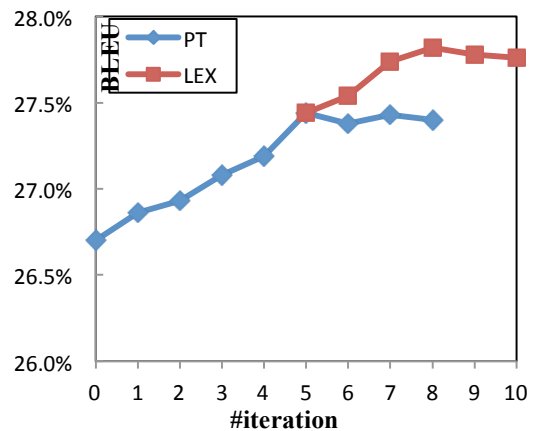


Figure 3. BLEU scores on the validation set as a function of the GT training iteration in two-stage training of both the phrase translation models (PT) and the lexicon models (LEX). The BLEU scores on training phrase models are shown in blue, and on training lexicon models in red.

### 5.3 Experiments on the IWSLT2011 benchmark

As the second evaluation task, we apply our new method described in this paper to the 2011 IWSLT Chinese-to-English machine translation benchmark (Federico et al., 2011). The main focus of the IWSLT2011 Evaluation is the translation of TED talks ([www.ted.com](http://www.ted.com)). These talks are originally given in English. In the Chinese-to-English translation task, we are provided with human translated Chinese text with punctuations inserted. The goal is to match the human transcribed English speech with punctuations.

This is an open-domain spoken language translation task. The training data consist of 110K sentences in the transcripts of the TED talks and their translations, in English and Chinese, respectively. Each sentence consists of 20 words on average. Two development sets are provided, namely, dev2010 and tst2010. They consist of 934 sentences and 1664 sentences, respectively. We use dev2010 for  $\lambda$  tuning and tst2010 for validation. The test set tst2011 consists of 1450 sentences.

In our system, a primary phrase table is trained from the 110K TED parallel training data, and a 3-gram LM is trained on the English side of the parallel data. We are also provided additional out-of-domain data for potential usage. From them, we train a secondary 5-gram LM on 115M sentences of supplementary English data, and a secondary phrase table from 500K sentences selected from the supplementary UN corpus by the method proposed by Axelrod et al. (2011).

In carrying out the maximum expected BLEU training, we use 100-best list and tune the regularization factor to the optimal value of  $\tau = 1 \times 10^{-5}$ . We only train the parameters of the primary phrase table. The secondary phrase table and LM are excluded from the training process since the out-of-domain phrase table is less relevant to the TED translation task, and the large LM slows down the N-best generation process significantly.

At the end, we perform one final MERT to tune the relative weights with all features including the secondary phrase table and LM.

The translation results are presented in Table 4. The baseline is a phrase-based system with all features including the secondary phrase table and LM. The new system uses the same features except that the primary phrase table is discriminatively

trained using maximum expected-BLEU and GT optimization as described earlier in this paper. The results are obtained using the two-stage training schedule, including six iterations for training phrase translation models and two iterations for training lexicon translation models. The results in Table 4 show that the proposed method leads to an improvement of 1.2 BLEU point over the baseline. This gives the best single system result on this task.

BLEU (%)	Validation	Test
Baseline	11.48	14.68
Max expected BLEU training	12.39	15.92

Table 4. The translation results on IWSLT 2011 MT\_CE task.

## 6. Summary

The contributions of this work can be summarized as follows. First, we propose a new objective function (Eq. 9) for training of large-scale translation models, including phrase and lexicon models, with more parameters than all previous methods have attempted. The objective function consists of 1) the utility function of expected BLEU score, and 2) the regularization term taking the form of KL divergence in the parameter space. The expected BLEU score is closely linked to translation quality and the regularization is essential when many parameters are trained at scale. The importance of both is verified experimentally with the results presented in this paper.

Second, through non-trivial derivation, we show that the novel objective function of Eq. (9) is amenable to iterative GT updates, where each update is equipped with a closed-form formula.

Third, the new objective function and new optimization technique are successfully applied to two important machine translation tasks, with implementation issues resolved (e.g., training schedule and hyper-parameter tuning, etc.). The superior results clearly demonstrate the effectiveness of the proposed algorithm.

## Acknowledgments

The authors are grateful to Chris Quirk, Mei-Yuh Hwang, and Bowen Zhou for the assistance with the MT system and/or for the valuable discussions.

## References

- Amittai Axelrod, Xiaodong He, Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In Proc. of EMNLP, 2011.
- Necip Fazil Ayan, and Bonnie J. Dorr. Going. 2006. Beyond AER: an extensive analysis of word alignments and their impact on MT. In Proc. of COLING-ACL, 2006.
- Leonard Baum and J. A. Eagon. 1967. An inequality with applications to statistical prediction for functions of Markov processes and to a model of ecology, Bulletin of the American Mathematical Society, Jan. 1967.
- Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In Proc. of ACL 2008.
- Peter F. Brown, Stephen A. Della Pietra, Vincent Della J. Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics, 1993.
- David Chiang, Steve DeNeefe, Yee Seng Chan, and Hwee Tou Ng, 2008. Decomposability of translation metrics for improved evaluation and efficient algorithms. In Proc. of EMNLP, 2008.
- David Chiang, Kevin Knight and Weri Wang, 2009. 11,001 new features for statistical machine translation. In Proc. of NAACL-HLT, 2009.
- Marcello Federico, L. Bentivogli, M. Paul, and S. Stueker. 2011. Overview of the IWSLT 2011 Evaluation Campaign. In Proc. of IWSLT, 2011.
- George Foster, Cyril Goutte and Roland Kuhn. 2010. Discriminative Instance Weighting for Domain Adaptation in Statistical Machine Translation. In Proc. of EMNLP, 2010.
- P. S. Gopalakrishnan, Dimitri Kanevsky, Arthur Nadas, and David Nahamoo. 1991. An inequality for rational functions with applications to some statistical estimation problems. IEEE Trans. Inform. Theory, 1991.
- Xiaodong He. 2007. Using Word-Dependent Transition Models in HMM based Word Alignment for Statistical Machine Translation. In *Proc. of the Second ACL Workshop on Statistical Machine Translation*.
- Xiaodong He, Li Deng, Wu Chou, 2008. Discriminative learning in sequential pattern recognition. IEEE Signal Processing Magazine, Sept. 2008.
- Philipp Koehn. 2002. Europarl: A Multilingual Corpus for Evaluation of Machine Translation.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase based translation. In Proc. of NAACL. 2003.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In Proc. of EMNLP 2004.
- Percy Liang, Alexandre Bouchard-Cote, Dan Klein and Ben. Taskar. 2006. An end-to-end discriminative approach to machine translation, In Proc. of COLING-ACL, 2006.
- Wolfgang Macherey, Franz Josef Och, gnacio Thayer, and Jakob Uskoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In Proc. of EMNLP 2008.
- Robert Moore and Chris Quirk. 2007. Faster Beam-Search Decoding for Phrasal Statistical Machine Translation. In Proc. of MT Summit XI.
- Franz Josef Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation, In Proc. of ACL 2002.
- Franz Josef Och, 2003, Minimum error rate training in statistical machine translation. In Proc. of ACL 2003.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In Proc. of ACL 2002.
- Daniel Povey. 2004. Discriminative Training for large Vocabulary Speech Recognition. Ph.D. dissertation, Cambridge University, Cambridge, UK, 2004.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In Proc. of ACL 2005.



- Antti-Veikko Rosti, Bing hang, Spyros Matsoukas, and Richard Schard Schwartz. 2011. Expected BLEU training for graphs: bbn system description for WMT system combination task. In Proc. of workshop on statistical machine translation 2011.
- David A Smith, Jason Eisner. 2006. Minimum risk annealing for training log-linear models, In Proc. of COLING-ACL 2006.
- Joern Wuebker, Arne Mauser and Hermann Ney. 2010. Training phrase translation models with leaving-one-out, In Proc. of ACL 2010.
- Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice Minimum Bayes-Risk decoding for statistical machine translation. In Proc. of EMNLP 2008.
- Xinyan Xiao, Yang Liu, Qun Liu, and Shouxun Lin. 2011. Fast Generation of Translation Forest for Large-Scale SMT Discriminative Training. In Proc. Of EMNLP 2011.

# Learning Translation Consensus with Structured Label Propagation

†Shujie Liu\*,

† Harbin Institute of Technology

Harbin, China

shujieliu@mtlab.hit.edu.cn

‡Chi-Ho Li, ‡Mu Li and ‡Ming Zhou

‡Microsoft Research Asia

Beijing, China

{chl, muli, mingzhou}@microsoft.com

## Abstract

In this paper, we address the issue for learning better translation consensus in machine translation (MT) research, and explore the search of translation consensus from similar, rather than the same, source sentences or their spans. Unlike previous work on this topic, we formulate the problem as structured labeling over a much smaller graph, and we propose a novel structured label propagation for the task. We convert such graph-based translation consensus from similar source strings into useful features both for n-best output re-ranking and for decoding algorithm. Experimental results show that, our method can significantly improve machine translation performance on both IWSLT and NIST data, compared with a state-of-the-art baseline.

## 1 Introduction

Consensus in translation has gained more and more attention in recent years. The principle of consensus can be sketched as “a translation candidate is deemed more plausible if it is supported by other translation candidates.” The actual formulation of the principle depends on whether the translation candidate is a complete sentence or just a span of it, whether the candidate is the same as or similar to the supporting candidates, and whether the supporting candidates come from the same or different MT system.

Translation consensus is employed in those minimum Bayes risk (MBR) approaches where the loss function of a translation is defined with respect to all other translation candidates. That is, the translation with the minimal Bayes risk is the one to the greatest extent similar to other candidates. These approaches include the work of Kumar and Byrne (2004), which re-ranks the n-best output of a MT decoder, and the work of Tromble et al. (2008) and Kumar et al. (2009), which does MBR decoding for lattices and hypergraphs.

Others extend consensus among translations from the same MT system to those from different MT systems. Collaborative decoding (Li et al., 2009) scores the translation of a source span by its n-gram similarity to the translations by other systems. Hypothesis mixture decoding (Duan et al., 2011) performs a second decoding process where the search space is enriched with new hypotheses composed out of existing hypotheses from multiple systems.

All these approaches are about utilizing consensus among translations for the same (span of) source sentence. It should be noted that consensus among translations of similar source sentences/spans is also helpful for good candidate selection. Consider the examples in Figure 1. For the source (Chinese) span “五百元以下的茶”, the MT system produced the correct translation for the second sentence, but it failed to do so for the first one. If the translation of the first sentence could take into consideration the translation of the second sentence, which is similar to but not exactly the same as the first one, the final translation output may be improved.

Following this line of reasoning, a discriminative learning method is proposed to constrain the translation of an input sentence using

---

\* This work has been done while the first author was visiting Microsoft Research Asia.

IWSLT Chinese to English Translation Task	
<b>Src</b>	你有没有五百元以下的茶?
<b>Ref</b>	<i>Do you have any tea under five hundred dollars ?</i>
<b>Best1</b>	<i>Do you have any less than five hundred dollars tea ?</i>
<b>Src</b>	我想要五百元以下的茶.
<b>Ref</b>	<i>I would like some tea under five hundred dollars .</i>
<b>Best1</b>	<i>I would like tea under five hundred dollars .</i>

Figure 1. Two sentences from IWSLT (Chinese to English) data set. "Src" stands for the source sentence, and "Ref" means the reference sentence. "Best1" is the final output of the decoder.

the most similar translation examples from translation memory (TM) systems (Ma et al., 2011). A classifier is applied to re-rank the n-best output of a decoder, taking as features the information about the agreement with those similar translation examples. Alexandrescu and Kirchhoff (2009) proposed a graph-based semi-supervised model to re-rank n-best translation output. Note that these two attempts are about translation consensus for similar sentences, and about re-ranking of n-best output. It is still an open question whether translation consensus for similar sentences/spans can be applied to the decoding process. Moreover, the method in Alexandrescu and Kirchhoff (2009) is formulated as a typical and simple label propagation, which leads to very large graph, thus making learning and search inefficient. (c.f. Section 3.)

In this paper, we attempt to leverage translation consensus among similar (spans of) source sentences in bilingual training data, by a novel graph-based model of translation consensus. Unlike Alexandrescu and Kirchhoff (2009), we reformulate the task of seeking translation consensus among source sentences as structured labeling. We propose a novel label propagation algorithm for structured labeling, which is much more efficient than simple label propagation, and derive useful MT decoder features out of it. We conduct experiments with IWSLT and NIST data, and experimental results show that, our method

can improve the translation performance significantly on both data sets, compared with a state-of-the-art baseline.

## 2 Graph-based Translation Consensus

Our MT system with graph-based translation consensus adopts the conventional log-linear model. For the source string  $f$ , the conditional probability of a translation candidate  $e$  is defined as:

$$p(e|f) = \frac{\exp(\sum_i(\lambda_i\psi_i(e, f)))}{\sum_{e' \in H(f)}(\exp(\sum_i(\lambda_i\psi_i(e', f))))} \quad (1)$$

where  $\psi$  is the feature vector,  $\lambda$  is the feature weights, and  $H(f)$  is the set of translation hypotheses in the search space.

Based on the commonly used features, two kinds of feature are added to equation (1), one is graph-based consensus features, which are about consensus among the translations of *similar* sentences/spans; the other is local consensus features, which are about consensus among the translations of the *same* sentence/span. We develop a structured label propagation method, which can calculate consensus statistics from translation candidates of similar source sentences/spans.

In the following, we explain why the standard, simple label propagation is not suitable for translation consensus, and then introduce how the problem is formulated as an instance of structured labeling, with the proposed structured label propagation algorithm, in section 3. Before elaborating how the graph model of consensus is constructed for both a decoder and N-best output re-ranking in section 5, we will describe how the consensus features and their feature weights can be trained in a semi-supervised way, in section 4.

## 3 Graph-based Structured Learning

In general, a graph-based model assigns labels to instances by considering the labels of similar instances. A graph is constructed so that each instance is represented by a node, and the weight of the edge between a pair of nodes represents the similarity between them. The gist of graph-based model is that, if two instances are connected by a strong edge, then their labels tend to be the same (Zhu, 2005).

In MT, the instances are source sentences or spans of source sentences, and the possible labels are their translation candidates. This scenario differs from the general case of graph-based model in two aspects. First, there are an indefinite, or even intractable, number of labels. Each of them is a string of words rather than a simple category. In the following we will call these labels as structured labels (Berlett et al., 2004). Second, labels are highly ‘instance-dependent’. In most cases, for any two different (spans of) source sentences, however small their difference is, their correct labels (translations) are not exactly the same. Therefore, the principle of graph-based translation consensus must be reformulated as, if two instances (source spans) are similar, then their labels (translations) tend to be similar (rather than the same).

Note that Alexandrescu and Kirchhoff (2009) do not consider translation as structured labeling. In their graph, a node does not represent only a source sentence but a pair of source sentence and its candidate translation, and there are only two possible labels for each node, namely, 1 (this is a good translation pair) and 0 (this is not a good translation pair). Thus their graph-based model is a normal example of the general graph-based model. The biggest problem of such a perspective is inefficiency. An average MT decoder considers a vast amount of translation candidates for each source sentence, and therefore the corresponding graph also contains a vast amount of nodes, thus rendering learning over a large dataset is infeasible.

### 3.1 Label Propagation for General Graph-based Models

A general graph-based model is iteratively trained by label propagation, in which  $p_{i,l}$ , the probability of label  $l$  for the node  $i$ , is updated with respect to the corresponding probabilities for  $i$ ’s neighboring nodes  $N(i)$ . In Zhu (2005), the updating rule is expressed in a matrix calculation. For convenience, the updating rule is expressed for each label here:

$$p_{i,l}^{t+1} = \sum_{j \in N(i)} T(i,j) p_{j,l}^t \quad (2)$$

where  $T(i,j)$ , the propagating probability, is defined as:

$$T(i,j) = \frac{w_{i,j}}{\sum_{j' \in N(i)} w_{i,j'}} \quad (3)$$

$w_{i,j}$  defines the weight of the edge, which is a similarity measure between nodes  $i$  and  $j$ .

Note that the graph contains nodes for training instances, whose correct labels are known. The probability of the correct label to each training instance is reset to 1 at the end of each iteration. With a suitable measure of instance/node similarity, it is expected that an unlabeled instance/node will find the most suitable label from similar labeled nodes.

### 3.2 Structured Label Propagation for Graph-based Learning

In structured learning like MT, different instances would not have the same correct label, and so the updating rule (2) is no longer valid, as the value of  $p_{i,l}$  should not be calculated based on  $p_{j,l}$ . Here we need a new updating rule so that  $p_{i,l}$  can be updated with respect to  $p_{j,l'}$ , where in general  $l \neq l'$ .

Let us start with the model in Alexandrescu and Kirchhoff (2009). According to them, a node in the graph represents the pair of some source sentence/span  $f$  and its translation candidate  $e$ . The updating rule (for the label 1 or 0) is:

$$p_{(f,e)}^{t+1} = \sum_{(f',e') \in NP(f,e)} T((f,e), (f',e')) p_{(f',e')}^t \quad (4)$$

where  $NP(f,e)$  is the set of neighbors of the node  $(f,e)$ .

When the problem is reformulated as structured labeling, each node represents the source sentence/span only, and the translation candidates become labels. The propagating probability  $T((f,e), (f',e'))$  has to be reformulated accordingly. A natural way is to decompose it into a component for nodes and a component for labels. Assuming that the two components are independent, then:

$$T((f,e), (f',e')) = T_s(f,f') T_l(e,e') \quad (5)$$

where  $T_s(f,f')$  is the propagating probability from source sentence/span  $f'$  to  $f$ , and  $T_l(e,e')$  is that from translation candidate  $e'$  to  $e$ .

The set of neighbors  $NP(f,e)$  of a pair  $(f,e)$  has also to be reformulated in terms of the set of neighbors  $N(f)$  of a source sentence/span  $f$ :

$$NP(f,e) = \{(f',e') | f' \in N(f), e' \in H(f')\} \quad (6)$$

where  $H(f')$  is the set of translation candidates for source  $f'$ . The new updating rule will then be:

$$\begin{aligned} p_{f,e}^{t+1} &= \sum_{f' \in N(f), e' \in H(f')} T_s(f, f') T_l(e, e') p_{f',e'}^t \\ &= \sum_{f' \in N(f)} \sum_{e' \in H(f')} T_s(f, f') T_l(e, e') p_{f',e'}^t \\ &= \sum_{f' \in N(f)} T_s(f, f') \sum_{e' \in H(f')} T_l(e, e') p_{f',e'}^t \quad (7) \end{aligned}$$

The new rule updates the probability of a translation  $e$  of a source sentence/span  $f$  with probabilities of similar translations  $e'$ 's of some similar source sentences/spans  $f'$ 's.

Propagation probability  $T_s(f, f')$  is as defined in equation (3), and  $T_l(e, e')$  is defined given some similarity measure  $sim(e, e')$  between labels  $e$  and  $e'$ :

$$T_l(e, e') = \frac{sim(e, e')}{\sum_{e'' \in H(f')} sim(e, e'')} \quad (8)$$

Note that rule (2) is a special case of rule (7), when  $sim(e, e')$  is defined as:

$$sim(e, e') = \begin{cases} 1 & \text{if } e = e'; \\ 0 & \text{otherwise;} \end{cases}$$

## 4 Features and Training

The last section sketched the structured label propagation algorithm. Before elaborating the details of how the actual graph is constructed, we would like to first introduce how the graph-based translation consensus can be used in an MT system.

### 4.1 Graph-based Consensus Features

The probability as estimated in equation (7) is taken as a group of new features in either a decoder or an n-best output re-ranker. We will call these features collectively as graph-based consensus features (GC):

$$\begin{aligned} GC(e, f) &= \\ \log \left( \sum_{f' \in N(f)} T_s(f, f') \sum_{e' \in H(f')} T_l(e, e') p_{f',e'}^t \right) \end{aligned} \quad (9)$$

Recall that,  $N(f)$  refers to source sentences/spans which are similar with  $f$ , and  $H(f')$  refers to

translation candidates of  $f'$ .  $p_{f',e'}$  is initialized with the translation posterior of  $e'$  given  $f'$ . The translation posterior is normalized in the n-best list. For the nodes representing the training sentence pairs, this posterior is fixed.  $T_l(e, e')$  is the propagating probability in equation (8), with the similarity measure  $sim(e, e')$  defined as the Dice co-efficient over the set of all n-grams in  $e$  and those in  $e'$ . That is,

$$sim(e, e') = Dice(NGr_n(e), NGr_n(e'))$$

where  $NGr_n(x)$  is the set of  $n$ -grams in string  $x$ , and  $Dice(A, B)$  is the Dice co-efficient over sets  $A$  and  $B$ :

$$Dice(A, B) = \frac{2|A \cap B|}{|A| + |B|}$$

We take  $1 \leq n \leq 4$  for similarity between translation candidates, thus leading to four features. The other propagating probability  $T_s(f, f')$ , as defined in equation (3), takes symmetrical sentence level BLEU as similarity measure<sup>1</sup>:

$$w_{f,f'} = \frac{1}{2} (BLEU_{sent}(f, f') + BLEU_{sent}(f', f))$$

where  $BLEU_{sent}(f, f')$  is defined as follows (Liang et al., 2006):

$$BLEU_{sent}(f, f') = \sum_{i=1}^4 \frac{i - BLEU(f, f')}{2^{4-i+1}} \quad (10)$$

where  $i - BLEU(f, f')$  is the IBM BLEU score computed over  $i$ -grams for hypothesis  $f$  using  $f'$  as reference.

In theory we could use other similarity measures such as edit distance, string kernel. Here simple n-gram similarity is used for the sake of efficiency.

### 4.2 Other Features

In addition to graph-based consensus features, we also propose local consensus features, defined over the n-best translation candidates as:

$$LC(e, f) = \log \left( \sum_{e' \in H(f)} p(e'|f) T_l(e, e') \right) \quad (11)$$

<sup>1</sup> BLEU is not symmetric, which means, different scores are obtained depending on which one is reference and which one is hypothesis.

where  $p(e'|f)$  is translation posterior. Like  $GC$ , there are four features with respect to the value of  $n$  in  $n$ -gram similarity measure.

We also use other fundamental features, such as translation probabilities, lexical weights, distortion probability, word penalty, and language model probability.

### 4.3 Training Method

When graph-based consensus is applied to an MT system, the graph will have nodes for training data, development (dev) data, and test data (details in Section 5). There is only one label/translation for each training data node. For each dev/test data node, the possible labels are the  $n$ -best translation candidates from the decoder. Note that there is mutual dependence between the consensus graph and the decoder. On the one hand, the MT decoder depends on the graph for the  $GC$  features. On the other hand, the graph needs the decoder to provide the translation candidates as possible labels, and their posterior probabilities as initial values of various  $p_{f,e}$ . Therefore, we can alternatively update graph-based consensus features and feature weights in the log-linear model.

---

#### Algorithm 1 Semi-Supervised Learning

---

```

 $GC^0 = 0;$ 
 $\lambda^t = \text{MERT}(S^{dev}, T^{dev}, GC^0);$ 
while not converged do
   $G^t = \text{CreatG}(S^{train}, T^{train}, S^{dev}, S^{test}, \lambda^t).$ 
   $GC^{t+1} = \text{StructLP}(G^t).$ 
   $\lambda^{t+1} = \text{MERT}(S^{dev}, T^{dev}, GC^{t+1})$ 
end while
return last  $(GC^t, \lambda^t)$ 

```

---

Algorithm 1 outlines our semi-supervised method for such alternative training. The entire process starts with a decoder without consensus features. Then a graph is constructed out of all training, dev, and test data. The subsequent structured label propagation provides  $GC$  feature values to the MT decoder. The decoder then adds the new features and re-trains all the feature weights by Minimum Error Rate Training (MERT) (Och, 2003). The decoder with new feature weights then provides new  $n$ -best candidates and their posteriors for constructing another consensus graph, which in turn gives rise to next round of

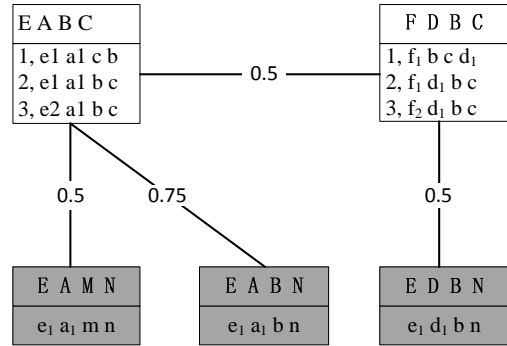


Figure 2. A toy graph constructed for re-ranking.

MERT. This alternation of structured label propagation and MERT stops when the BLEU score on dev data converges, or a pre-set limit (10 rounds) is reached.

## 5 Graph Construction

A technical detail is still needed to complete the description of graph-based consensus, namely, how the actual consensus graph is constructed. We will divide the discussion into two sections regarding how the graph is used.

### 5.1 Graph Construction for Re-Ranking

When graph-based consensus is used for re-ranking the  $n$ -best outputs of a decoder, each node in the graph corresponds to a complete sentence. A separate node is created for each source sentence in training data, dev data, and test data. For any node from training data (henceforth training node), it is labeled with the correct translation, and  $p_{f,e}$  is fixed as 1. If there are sentence pairs with the same source sentence but different translations, all the translations will be assigned as labels to that source sentence, and the corresponding probabilities are estimated by MLE. There is no edge between training nodes, since we suppose all the sentences of the training data are correct, and it is pointless to re-estimate the confidence of those sentence pairs.

Each node from dev/test data (henceforth test node) is unlabeled, but it will be given an  $n$ -best list of translation candidates as possible labels from a MT decoder. The decoder also provides translation posteriors as the initial confidences of

the labels. A test node can be connected to training nodes and other test nodes. If the source sentences of a test node and some other node are sufficiently similar, a similarity edge is created between them. In our experiment we measure similarity by symmetrical sentence level BLEU of source sentences, and 0.3 is taken as the threshold for edge creation.

Figure 2 shows a toy example graph. Each node is depicted as rectangle with the upper half showing the source sentence and the lower half showing the correct or possible labels. Training nodes are in grey while test nodes are in white. The edges between the nodes are weighted by the similarities between the corresponding source sentences.

## 5.2 Graph Construction for Decoding

Graph-based consensus can also be used in the decoding algorithm, by re-ranking the translation candidates of not only the entire source sentence but also every source span. Accordingly the graph does not contain only the nodes for source sentences but also the nodes for all source spans. It is needed to find the candidate labels for each source span.

It is not difficult to handle test nodes, since the purpose of MT decoder is to get all possible segmentations of a source sentence in dev/test data, search for the translation candidates of each source span, and calculate the probabilities of the candidates. Therefore, the cells in the search space of a decoder can be directly mapped as test nodes in the graph.

Training nodes can be handled similarly, by applying forced alignment. Forced alignment performs phrase segmentation and alignment of each sentence pair of the training data using the full translation system as in decoding (Wuebker et al., 2010). In simpler term, for each sentence pair in training data, a decoder is applied to the source side, and all the translation candidates that do not match any substring of the target side are deleted. The cells of in such a reduced search space of the decoder can be directly mapped as training nodes in the graph, just as in the case of test nodes. Note that, due to pruning in both decoding and translation model training, forced alignment may fail, i.e. the decoder may not be able to produce target side of a sentence pair. In such case we still map the cells in the search space as training nodes.

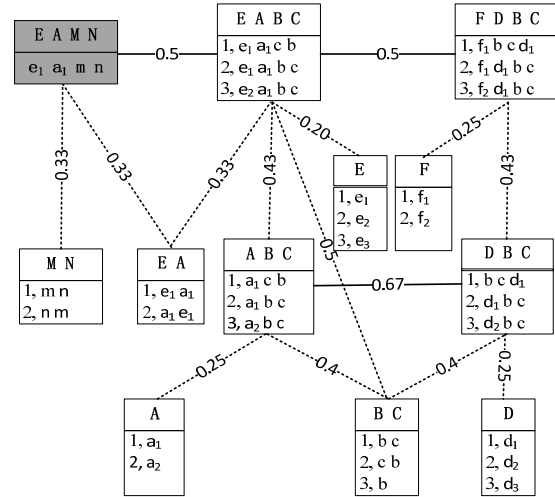


Figure 3. A toy example graph for decoding. Edges in dash line indicate relation between a span and its sub-span, whereas edges of solid line indicate source side similarity.

Note also that the shorter a source span is, the more likely it appears in more than one source sentence. All the translation candidates of the same source span in different source sentences are merged.

Edge creation is the same as that in graph construction for n-best re-ranking, except that two nodes are always connected if they are about a span and its sub-span. This exception ensures that shorter spans can always receive propagation from longer ones, and vice versa.

Figure 3 shows a toy example. There is one node for the training sentence "E A M N" and two nodes for the test sentences "E A B C" and "F D B C". All the other nodes represent spans. The node "M N" and "E A" are created according to the forced alignment result of the sentence "E A M N". As we see, the translation candidates for "M N" and "E A" are not the sub-strings from the target sentence of "E A M N". There are two kinds of edges. Dash lines are edges connecting nodes of a span and its sub-span, such as the one between "E A B C" and "E". Solid lines are edges connecting nodes with sufficient source side n-gram similarity, such as the one between "E A M N" and "E A B C".

## 6 Experiments and Results

In this section, graph-based translation consensus is tested on the Chinese to English translation tasks. The evaluation method is the case insensitive IBM BLEU-4 (Papineni et al., 2002). Significant testing is carried out using bootstrap re-sampling method proposed by Koehn (2004) with a 95% confidence level.

### 6.1 Experimental Data Setting and Baselines

We test our method with two data settings: one is IWSLT data set, the other is NIST data set. Our baseline decoder is an in-house implementation of Bracketing Transduction Grammar (Dekai Wu, 1997) (BTG) in CKY-style decoding with a lexical reordering model trained with maximum entropy (Xiong et al., 2006). The features we used are commonly used features as standard BTG decoder, such as translation probabilities, lexical weights, language model, word penalty and distortion probabilities.

Our IWSLT data is the IWSLT 2009 dialog task data set. The training data include the BTEC and SLDB training data. The training data contains 81k sentence pairs, 655k Chinese words and 806 English words. The language model is 5-gram language model trained with the target sentences in the training data. The test set is devset9, and the development set for MERT comprises both devset8 and the Chinese DIALOG set. The baseline results on IWSLT data are shown in Table 1.

	devset8+dialog	devset9
Baseline	48.79	44.73

Table 1. Baselines for IWSLT data

For the NIST data set, the bilingual training data we used is NIST 2008 training set excluding the Hong Kong Law and Hong Kong Hansard. The training data contains 354k sentence pairs, 8M Chinese words and 10M English words. The language model is 5-gram language model trained with the Giga-Word corpus plus the English sentences in the training data. The development data utilized to tune the feature weights of our decoder is NIST’03 evaluation set, and test sets are NIST’05 and NIST’08 evaluation sets. The baseline results on NIST data are shown in Table 2.

	NIST’03	NIST’05	NIST’08
Baseline	38.57	38.21	27.52

Table 2. Baselines for NIST data

### 6.2 Experimental Result

Table 3 shows the performance of our consensus-based re-ranking and decoding on the IWSLT data set. To perform consensus-based re-ranking, we first use the baseline decoder to get the n-best list for each sentence of development and test data, then we create graph using the n-best lists and training data as we described in section 5.1, and perform semi-supervised training as mentioned in section 4.3. As we can see from Table 3, our consensus-based re-ranking (G-Re-Rank) outperforms the baseline significantly, not only for the development data, but also for the test data.

Instead of using graph-based consensus confidence as features in the log-linear model, we perform structured label propagation (Struct-LP) to re-rank the n-best list directly, and the similarity measures for source sentences and translation candidates are symmetrical sentence level BLEU (equation (10)). Using Struct-LP, the performance is significantly improved, compared with the baseline, but not as well as G-Re-Rank.

	devset8+dialog	devset9
Baseline	48.79	44.73
Struct-LP	<b>49.86</b>	<b>45.54</b>
G-Re-Rank	<b>50.66</b>	<b>46.52</b>
G-Re-Rank-GC	<b>50.23</b>	<b>45.96</b>
G-Re-Rank-LC	<b>49.87</b>	<b>45.84</b>
G-Decode	<b>51.20</b>	<b>47.31</b>
G-Decode-GC	<b>50.46</b>	<b>46.21</b>
G-Decode-LC	<b>50.11</b>	<b>46.17</b>

Table 3. Consensus-based re-ranking and decoding for IWSLT data set. The results in bold type are significantly better than the baseline.

We use the baseline system to perform forced alignment procedure on the training data, and create span nodes using the derivation tree of the forced alignment. We also saved the spans of the sentences from development and test data, which will be used to create the responding nodes for consensus-based decoding. In such a way, we create the graph for decoding, and perform semi-



supervised training to calculate graph-based consensus features, and tune the weights for all the features we used. In Table 3, we can see that our consensus-based decoding (G-Decode) is much better than baseline, and also better than consensus-based re-ranking method. That is reasonable since the neighbor/local similarity features not only re-rank the final n-best output, but also the spans during decoding.

To test the contribution of each kind of features, we first remove all the local consensus features and perform consensus-based re-ranking and decoding (G-Re-Rank-GC and G-Decode-GC), and then we remove all the graph-based consensus features to test the contribution of local consensus features (G-Re-Rank-LC and G-Decode-LC). Without the graph-based consensus features, our consensus-based re-ranking and decoding is simplified into a consensus re-ranking and consensus decoding system, which only re-rank the candidates according to the consensus information of other candidates in the same n-best list.

From Table 3, we can see, the G-Re-Rank-LC and G-Decode-LC improve the performance of development data and test data, but not as much as G-Re-Rank and G-Decode do. G-Re-Rank-GC and G-Decode-GC improve the performance of machine translation according to the baseline. G-Re-Rank-GC does not achieve the same performance as G-Re-Rank-LC does. Compared with G-Decode-LC, the performance with G-Decode-GC is much better.

	NIST'03	NIST'05	NIST'08
Baseline	38.57	38.21	27.52
Struct-LP	38.79	38.52	28.06
G-Re-Rank	<b>39.21</b>	<b>38.93</b>	<b>28.18</b>
G-Re-Rank-GC	38.92	38.76	<b>28.21</b>
G-Re-Rank-LC	38.90	38.65	27.88
G-Decode	<b>39.62</b>	<b>39.17</b>	<b>28.76</b>
G-Decode-GC	<b>39.42</b>	<b>39.02</b>	<b>28.51</b>
G-Decode-LC	39.17	38.70	<b>28.20</b>

Table 4. Consensus-based re-ranking and decoding for NIST data set. The results in bold type are significantly better than the baseline.

We also conduct experiments on NIST data, and results are shown in Table 4. The consensus-based

re-ranking methods are performed in the same way as for IWSLT data, but for consensus-based decoding, the data set contains too many sentence pairs to be held in one graph for our machine. We apply the method of Alexandrescu and Kirchhoff (2009) to construct separate graphs for each development and test sentence without losing global connectivity information. We perform modified label propagation with the separate graphs to get the graph-based consensus for n-best list of each sentence, and the graph-based consensus will be recorded for the MERT to tune the weights.

From Table 4, we can see that, Struct-LP improves the performance slightly, but not significantly. Local consensus features (G-Re-Rank-LC and G-Decode-LC) improve the performance slightly. The combination of graph-based and local consensus features can improve the translation performance significantly on SMT re-ranking. With graph-based consensus features, G-Decode-GC achieves significant performance gain, and combined with local consensus features, G-Decode performance is improved farther.

## 7 Conclusion and Future Work

In this paper, we extend the consensus method by collecting consensus statistics, not only from translation candidates of the same source sentence/span, but also from those of similar ones. To calculate consensus statistics, we develop a novel structured label propagation method for structured learning problems, such as machine translation. Note that, the structured label propagation can be applied to other structured learning tasks, such as POS tagging and syntactic parsing. The consensus statistics are integrated into the conventional log-linear model as features. The features and weights are tuned with an iterative semi-supervised method. We conduct experiments on IWSLT and NIST data, and our method can improve the performance significantly.

In this paper, we only tried Dice co-efficient of n-grams and symmetrical sentence level BLEU as similarity measures. In the future, we will explore other consensus features and other similarity measures, which may take document level information, or syntactic and semantic information into consideration. We also plan to introduce feature to model the similarity of the source

sentences, which are reflected by only one score in our paper, and optimize the parameters with CRF model.

## References

- Andrei Alexandrescu, Katrin Kirchhoff. 2009. Graph-based learning for statistical machine translation. In *Proceedings of Human Language Technologies and Annual Conference of the North American Chapter of the ACL*, pages 119-127.
- Peter L. Bertlett, Michael Collins, Ben Taskar and David McAllester. 2004. Exponentiated gradient algorithms for large-margin structured classification. In *Proceedings of Advances in Neural Information Processing Systems*.
- John DeNero, David Chiang, and Kevin Knight. 2009. Fast consensus decoding over translation forests. In *Proceedings of the Association for Computational Linguistics*, pages 567-575.
- John DeNero, Shankar Kumar, Ciprian Chelba and Franz Och. 2010. Model combination for machine translation. In *Proceedings of the North American Association for Computational Linguistics*, pages 975-983.
- Nan Duan, Mu Li, Dongdong Zhang, and Ming Zhou. 2010. Mixture model-based minimum bayes risk decoding using multiple machine translation Systems. In *Proceedings of the International Conference on Computational Linguistics*, pages 313-321.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 388-395.
- Shankar Kumar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the North American Association for Computational Linguistics*, pages 169-176.
- Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Association for Computational Linguistics*, pages 163-171.
- Mu Li, Nan Duan, Dongdong Zhang, Chi-Ho Li, and Ming Zhou. 2009. Collaborative decoding: partial hypothesis re-ranking using translation consensus between decoders. In *Proceedings of the Association for Computational Linguistics*, pages 585-592.
- Percy Liang, Alexandre Bouchard-Cote, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the International Conference on Computational Linguistics and the ACL*, pages 761-768
- YanJun Ma, Yifan He, Andy Way, Josef van Genabith. 2011. Consistent translation using discriminative learning: a translation memory-inspired approach. In *Proceedings of the Association for Computational Linguistics*, pages 1239-1248.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 160-167.
- Kishore Papineni, Salim Roukos, Todd Ward and Weijing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 311-318.
- Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing*, pages 620-629.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3).
- Joern Wuebker, Arne Mauser and Hermann Ney. 2010. Training phrase translation models with leaving-one-out. In *Proceedings of the Association for Computational Linguistics*, pages 475-484.
- Deyi Xiong, Qun Liu and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 521-528.
- Xiaojin Zhu. 2005. Semi-supervised learning with graphs. *Ph.D. thesis*, Carnegie Mellon University. CMU-LTI-05-192.

# Smaller Alignment Models for Better Translations: Unsupervised Word Alignment with the $\ell_0$ -norm

Ashish Vaswani    Liang Huang    David Chiang

University of Southern California

Information Sciences Institute

{avaswani, lhuang, chiang}@isi.edu

## Abstract

Two decades after their invention, the IBM word-based translation models, widely available in the GIZA++ toolkit, remain the dominant approach to word alignment and an integral part of many statistical translation systems. Although many models have surpassed them in accuracy, none have supplanted them in practice. In this paper, we propose a simple extension to the IBM models: an  $\ell_0$  prior to encourage sparsity in the word-to-word translation model. We explain how to implement this extension efficiently for large-scale data (also released as a modification to GIZA++) and demonstrate, in experiments on Czech, Arabic, Chinese, and Urdu to English translation, significant improvements over IBM Model 4 in both word alignment (up to +6.7 F1) and translation quality (up to +1.4 BLEU).

## 1 Introduction

Automatic word alignment is a vital component of nearly all current statistical translation pipelines. Although state-of-the-art translation models use rules that operate on units bigger than words (like phrases or tree fragments), they nearly always use word alignments to drive extraction of those translation rules. The dominant approach to word alignment has been the IBM models (Brown et al., 1993) together with the HMM model (Vogel et al., 1996). These models are unsupervised, making them applicable to any language pair for which parallel text is available. Moreover, they are widely disseminated in the open-source GIZA++ toolkit (Och and Ney, 2004). These properties make them the default choice for most statistical MT systems.

In the decades since their invention, many models have surpassed them in accuracy, but none has supplanted them in practice. Some of these models are partially supervised, combining unlabeled parallel text with manually-aligned parallel text (Moore, 2005; Taskar et al., 2005; Riesa and Marcu, 2010). Although manually-aligned data is very valuable, it is only available for a small number of language pairs. Other models are unsupervised like the IBM models (Liang et al., 2006; Graça et al., 2010; Dyer et al., 2011), but have not been as widely adopted as GIZA++ has.

In this paper, we propose a simple extension to the IBM/HMM models that is unsupervised like the IBM models, is as scalable as GIZA++ because it is implemented on top of GIZA++, and provides significant improvements in both alignment and translation quality. It extends the IBM/HMM models by incorporating an  $\ell_0$  prior, inspired by the principle of minimum description length (Barron et al., 1998), to encourage sparsity in the word-to-word translation model (Section 2.2). This extension follows our previous work on unsupervised part-of-speech tagging (Vaswani et al., 2010), but enables it to scale to the large datasets typical in word alignment, using an efficient training method based on projected gradient descent (Section 2.3). Experiments on Czech-, Arabic-, Chinese- and Urdu-English translation (Section 3) demonstrate consistent significant improvements over IBM Model 4 in both word alignment (up to +6.7 F1) and translation quality (up to +1.4 BLEU). Our implementation has been released as a simple modification to the GIZA++ toolkit that can be used as a drop-in replacement for GIZA++ in any existing MT pipeline.

## 2 Method

We start with a brief review of the IBM and HMM word alignment models, then describe how to extend them with a smoothed  $\ell_0$  prior and how to efficiently train them.

### 2.1 IBM Models and HMM

Given a French string  $\mathbf{f} = f_1 \cdots f_j \cdots f_m$  and an English string  $\mathbf{e} = e_1 \cdots e_i \cdots e_\ell$ , these models describe the process by which the French string is generated by the English string via the alignment  $\mathbf{a} = a_1, \dots, a_j, \dots, a_m$ . Each  $a_j$  is a hidden variable, indicating which English word  $e_{a_j}$  the French word  $f_j$  is aligned to.

In IBM Model 1–2 and the HMM model, the joint probability of the French sentence and alignment given the English sentence is

$$P(\mathbf{f}, \mathbf{a} | \mathbf{e}) = \prod_{j=1}^m d(a_j | a_{j-1}, j) t(f_j | e_{a_j}). \quad (1)$$

The parameters of these models are the distortion probabilities  $d(a_j | a_{j-1}, j)$  and the translation probabilities  $t(f_j | e_{a_j})$ . The three models differ in their estimation of  $d$ , but the differences do not concern us here. All three models, as well as IBM Models 3–5, share the same  $t$ . For further details of these models, the reader is referred to the original papers describing them (Brown et al., 1993; Vogel et al., 1996).

Let  $\theta$  stand for all the parameters of the model. The standard training procedure is to find the parameter values that maximize the likelihood, or, equivalently, minimize the negative log-likelihood of the observed data:

$$\hat{\theta} = \arg \min_{\theta} (-\log P(\mathbf{f} | \mathbf{e}, \theta)) \quad (2)$$

$$= \arg \min_{\theta} \left( -\log \sum_{\mathbf{a}} P(\mathbf{f}, \mathbf{a} | \mathbf{e}, \theta) \right) \quad (3)$$

This is done using the Expectation-Maximization (EM) algorithm (Dempster et al., 1977).

### 2.2 MAP-EM with the $\ell_0$ -norm

Maximum likelihood training is prone to overfitting, especially in models with many parameters. In word alignment, one well-known manifestation of overfitting is that rare words can act as “garbage collectors”

(Moore, 2004), aligning to many unrelated words. This hurts alignment precision and rule-extraction recall. Previous attempted remedies include early stopping, smoothing (Moore, 2004), and posterior regularization (Graça et al., 2010).

We have previously proposed another simple remedy to overfitting in the context of unsupervised part-of-speech tagging (Vaswani et al., 2010), which is to minimize the size of the model using a smoothed  $\ell_0$  prior. Applying this prior to an HMM improves tagging accuracy for both Italian and English.

Here, our goal is to apply a similar prior in a word-alignment model to the word-to-word translation probabilities  $t(f | e)$ . We leave the distortion models alone, since they are not very large, and there is not much reason to believe that we can profit from compacting them.

With the addition of the  $\ell_0$  prior, the MAP (maximum *a posteriori*) objective function is

$$\hat{\theta} = \arg \min_{\theta} (-\log P(\mathbf{f} | \mathbf{e}, \theta) P(\theta)) \quad (4)$$

where

$$P(\theta) \propto \exp(-\alpha \|\theta\|_0^\beta) \quad (5)$$

and

$$\|\theta\|_0^\beta = \sum_{e,f} \left( 1 - \exp\left(-\frac{t(f|e)}{\beta}\right) \right) \quad (6)$$

is a smoothed approximation of the  $\ell_0$ -norm. The hyperparameter  $\beta$  controls the tightness of the approximation, as illustrated in Figure 1. Substituting back into (4) and dropping constant terms, we get the following optimization problem: minimize

$$-\log P(\mathbf{f} | \mathbf{e}, \theta) - \alpha \sum_{e,f} \exp\left(-\frac{t(f|e)}{\beta}\right) \quad (7)$$

subject to the constraints

$$\sum_f t(f|e) = 1 \quad \text{for all } e. \quad (8)$$

We can carry out the optimization in (7) with the MAP-EM algorithm (Bishop, 2006). EM and MAP-EM share the same E-step; the difference lies in the

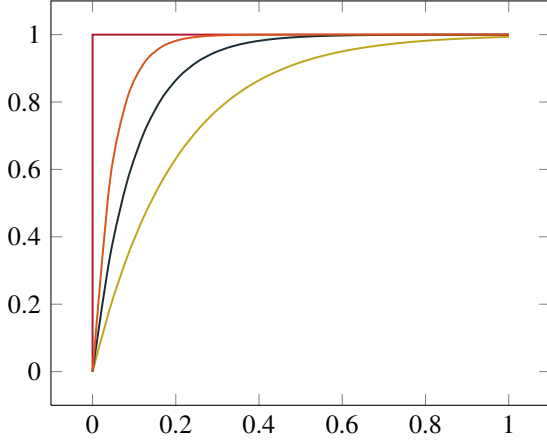


Figure 1: The  $\ell_0$ -norm (top curve) and smoothed approximations (below) for  $\beta = 0.05, 0.1, 0.2$ .

M-step. For vanilla EM, the M-step is:

$$\hat{\theta} = \arg \min_{\theta} \left( - \sum_{e,f} E[C(e, f)] \log t(f | e) \right) \quad (9)$$

again subject to the constraints (8). The count  $C(e, f)$  is the number of times that  $f$  occurs aligned to  $e$ . For MAP-EM, it is:

$$\hat{\theta} = \arg \min_{\theta} \left( - \sum_{e,f} E[C(e, f)] \log t(f | e) - \alpha \sum_{e,f} \exp \frac{-t(f | e)}{\beta} \right) \quad (10)$$

This optimization problem is non-convex, and we do not know of a closed-form solution. Previously (Vaswani et al., 2010), we used ALGENCAN, a non-linear optimization toolkit, but this solution does not scale well to the number of parameters involved in word alignment models. Instead, we use a simpler and more scalable method which we describe in the next section.

### 2.3 Projected gradient descent

Following Schoenemann (2011b), we use projected gradient descent (PGD) to solve the M-step (but with the  $\ell_0$ -norm instead of the  $\ell_1$ -norm). Gradient projection methods are attractive solutions to constrained optimization problems, particularly when the constraints on the parameters are simple (Bertsekas, 1999). Let  $F(\theta)$  be the objective function in

(10); we seek to minimize this function. As in previous work (Vaswani et al., 2010), we optimize each set of parameters  $\{t(\cdot | e)\}$  separately for each English word type  $e$ . The inputs to the PGD are the expected counts  $E[C(e, f)]$  and the current word-to-word conditional probabilities  $\theta$ . We run PGD for  $K$  iterations, producing a sequence of intermediate parameter vectors  $\theta^1, \dots, \theta^k, \dots, \theta^K$ . Each iteration has two steps, a projection step and a line search.

**Projection step** In this step, we compute:

$$\bar{\theta}^k = [\theta^k - s \nabla F(\theta^k)]^{\Delta} \quad (11)$$

This moves  $\theta$  in the direction of steepest descent ( $\nabla F$ ) with step size  $s$ , and then the function  $[\cdot]^{\Delta}$  projects the resulting point onto the simplex; that is, it finds the nearest point that satisfies the constraints (8).

The gradient  $\nabla F(\theta^k)$  is

$$\frac{\partial F}{\partial t(f | e)} = - \frac{E[C(f, e)]}{t(f | e)} + \frac{\alpha}{\beta} \exp \frac{-t(f | e)}{\beta} \quad (12)$$

In contrast to Schoenemann (2011b), we use an  $O(n \log n)$  algorithm for the projection step due to Duchi et al. (2008), shown in Pseudocode 1.

---

**Pseudocode 1** Project input vector  $\mathbf{u} \in \mathbb{R}^n$  onto the probability simplex.

---

```

 $\mathbf{v} = \mathbf{u}$  sorted in non-increasing order
 $\rho = 0$ 
for  $i = 1$  to  $n$  do
  if  $v_i - \frac{1}{i} (\sum_{r=1}^i v_r - 1) > 0$  then
     $\rho = i$ 
  end if
end for
 $\eta = \frac{1}{\rho} (\sum_{r=1}^{\rho} v_r - 1)$ 
 $w_r = \max\{v_r - \eta, 0\}$  for  $1 \leq r \leq n$ 
return  $\mathbf{w}$ 

```

---

**Line search** Next, we move to a point between  $\theta^k$  and  $\bar{\theta}^k$  that satisfies the *Armijo condition*,

$$F(\theta^k + \delta_m) \leq F(\theta^k) + \sigma (\nabla F(\theta^k) \cdot \delta_m) \quad (13)$$

where  $\delta_m = \gamma^m (\bar{\theta}^k - \theta^k)$  and  $\sigma$  and  $\gamma$  are both constants in  $(0, 1)$ . We try values  $m = 1, 2, \dots$  until the Armijo condition (13) is satisfied or the limit  $m = 20$

---

**Pseudocode 2** Find a point between  $\theta^k$  and  $\bar{\theta}^k$  that satisfies the Armijo condition.

---

```

 $F_{min} = F(\theta^k)$ 
 $\theta_{min} = \theta^k$ 
for  $m = 1$  to  $20$  do
   $\delta_m = \gamma^m (\bar{\theta}^k - \theta^k)$ 
  if  $F(\theta^k + \delta_m) < F_{min}$  then
     $F_{min} = F(\theta^k + \delta_m)$ 
     $\theta_{min} = \theta^k + \delta_m$ 
  end if
  if  $F(\theta^k + \delta_m) \leq F(\theta^k) + \sigma (\nabla F(\theta^k) \cdot \delta_m)$  then
    break
  end if
end for
 $\theta^{k+1} = \theta_{min}$ 
return  $\theta^{k+1}$ 

```

---

is reached. (Note that we don't allow  $m = 0$  because this can cause  $\theta^k + \delta_m$  to land on the boundary of the probability simplex, where the objective function is undefined.) Then we set  $\theta^{k+1}$  to the point in  $\{\theta^k\} \cup \{\theta^k + \delta_m \mid 1 \leq m \leq 20\}$  that minimizes  $F$ . The line search algorithm is summarized in Pseudocode 2.

In our implementation, we set  $\gamma = 0.5$  and  $\sigma = 0.5$ . We keep  $s$  fixed for all PGD iterations; we experimented with  $s \in \{0.1, 0.5\}$  and did not observe significant changes in F-score. We run the projection step and line search alternately for at most  $K$  iterations, terminating early if there is no change in  $\theta^k$  from one iteration to the next. We set  $K = 35$  for the large Arabic-English experiment; for all other conditions, we set  $K = 50$ . These choices were made to balance efficiency and accuracy. We found that values of  $K$  between 30 and 75 were generally reasonable.

### 3 Experiments

To demonstrate the effect of the  $\ell_0$ -norm on the IBM models, we performed experiments on four translation tasks: Arabic-English, Chinese-English, and Urdu-English from the NIST Open MT Evaluation, and the Czech-English translation from the Workshop on Machine Translation (WMT) shared task. We measured the accuracy of word alignments generated by GIZA++ with and without the  $\ell_0$ -norm,

and also translation accuracy of systems trained using the word alignments. Across all tests, we found strong improvements from adding the  $\ell_0$ -norm.

#### 3.1 Training

We have implemented our algorithm as an open-source extension to GIZA++.<sup>1</sup> Usage of the extension is identical to standard GIZA++, except that the user can switch the  $\ell_0$  prior on or off, and adjust the hyperparameters  $\alpha$  and  $\beta$ .

For vanilla EM, we ran five iterations of Model 1, five iterations of HMM, and ten iterations of Model 4. For our approach, we first ran one iteration of Model 1, followed by four iterations of Model 1 with smoothed  $\ell_0$ , followed by five iterations of HMM with smoothed  $\ell_0$ . Finally, we ran ten iterations of Model 4.<sup>2</sup>

We used the following parallel data:

- Chinese-English: selected data from the constrained task of the NIST 2009 Open MT Evaluation.<sup>3</sup>
- Arabic-English: all available data for the constrained track of NIST 2009, excluding United Nations proceedings (LDC2004E13), ISI Automatically Extracted Parallel Text (LDC2007E08), and Ummah newswire text (LDC2004T18), for a total of 5.4+4.3 million words. We also experimented on a larger Arabic-English parallel text of 44+37 million words from the DARPA GALE program.
- Urdu-English: all available data for the constrained track of NIST 2009.

---

<sup>1</sup>The code can be downloaded from the first author's website at <http://www.isi.edu/~avaswani/giza-pp-10.html>.

<sup>2</sup>GIZA++ allows changing some heuristic parameters for efficient training. Currently, we set two of these to zero: `mincountincrease` and `probcutoff`. In the default setting, both are set to  $10^{-7}$ . We set `probcutoff` to 0 because we would like the optimization to learn the parameter values. For a fair comparison, we applied the same setting to our vanilla EM training as well. To test, we ran GIZA++ with the default setting on the smaller of our two Arabic-English datasets with the same number of iterations and found no change in F-score.

<sup>3</sup>LDC catalog numbers LDC2003E07, LDC2003E14, LDC2005E83, LDC2005T06, LDC2006E24, LDC2006E34, LDC2006E85, LDC2006E86, LDC2006E92, and LDC2006E93.

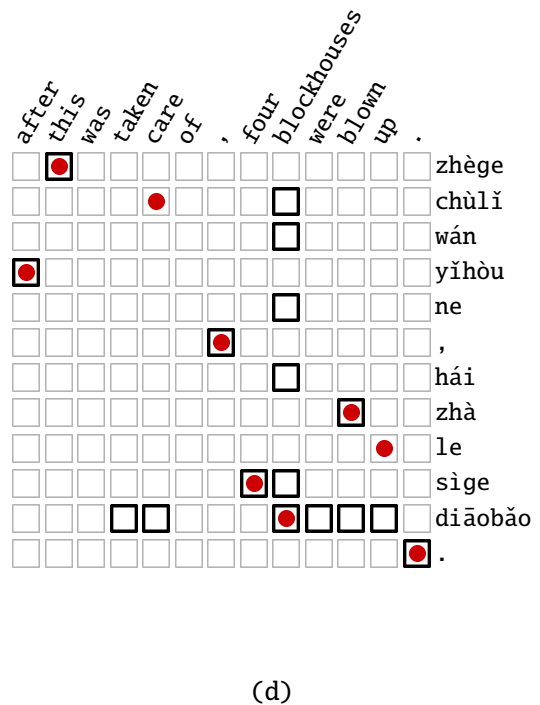
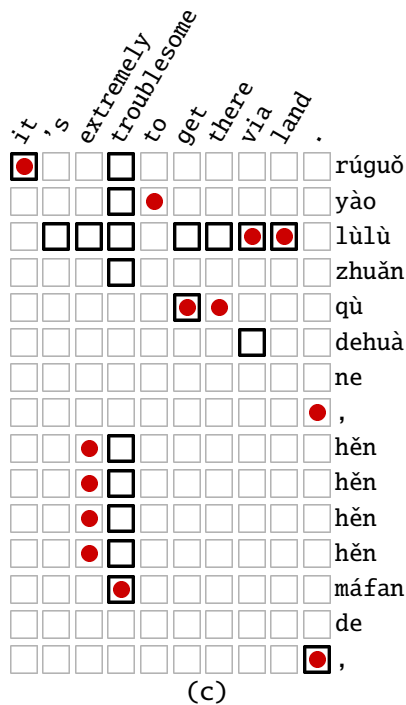
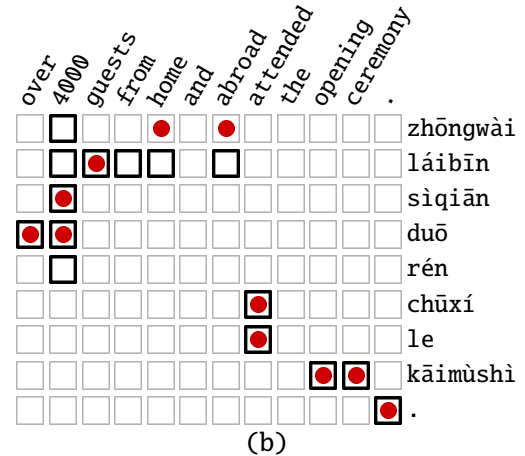
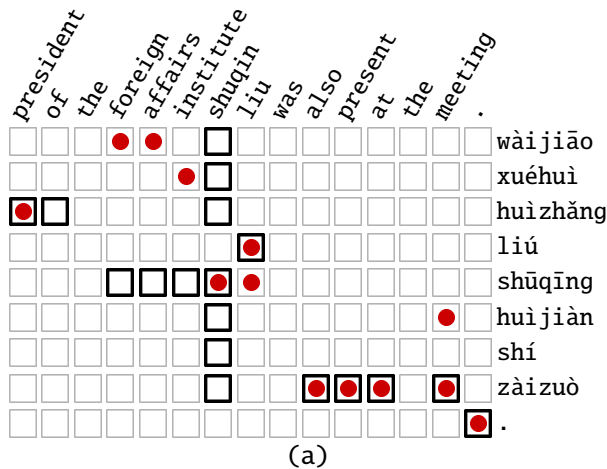


Figure 2: Smoothed- $\ell_0$  alignments (red circles) correct many errors in the baseline GIZA++ alignments (black squares), as shown in four Chinese-English examples (the red circles are almost perfect for these examples, except for minor mistakes such as liu-shūqīng and meeting-zàizuò in (a) and .-, in (c)). In particular, the baseline system demonstrates typical “garbage-collection” phenomena in proper name “shuqing” in both languages in (a), number “4000” and word “lái bīn” (lit. “guest”) in (b), word “troublesome” and “lù lù” (lit. “land-route”) in (c), and “blockhouses” and “diào bǎo” (lit. “bunker”) in (d). We found this garbage-collection behavior to be especially common with proper names, numbers, and uncommon words in both languages. Most interestingly, in (c), our smoothed- $\ell_0$  system correctly aligns “extremely” to “hěn hěn hěn hěn” (lit. “very very very very”) which is rare in the bitext.

task	data (M)	system	align F1 (%)	word trans (M)	$\tilde{\phi}_{sing.}$	B (%)		
						2008	2009	2010
Chi-Eng	9.6+12	baseline	73.2	3.5	6.2	28.7		
		$\ell_0$ -norm	76.5	2.0	3.3	29.5		
		difference	+3.3	-43%	-47%	+0.8		
Ara-Eng	5.4+4.3	baseline	65.0	3.1	4.5	39.8	42.5	
		$\ell_0$ -norm	70.8	1.8	1.8	41.1	43.7	
		difference	+5.9	-39%	-60%	+1.3	+1.2	
Ara-Eng	44+37	baseline	66.2	15	5.0	41.6	44.9	
		$\ell_0$ -norm	71.8	7.9	1.8	42.5	45.3	
		difference	+5.6	-47%	-64%	+0.9	+0.4	
Urd-Eng	1.7+1.5	baseline		1.7	4.5	25.3*	29.8	
		$\ell_0$ -norm		1.2	2.2	25.9*	31.2	
		difference		-29%	-51%	+0.6*	+1.4	
Cze-Eng	2.1+2.3	baseline	65.6	1.5	3.0		17.3	18.0
		$\ell_0$ -norm	72.3	1.0	1.4		17.9	18.4
		difference	+6.7	-33%	-53%		+0.6	+0.4

Table 1: Adding the  $\ell_0$ -norm to the IBM models improves both alignment and translation accuracy across four different language pairs. The *word trans* column also shows that the number of distinct word translations (i.e., the size of the lexical weighting table) is reduced. The  $\tilde{\phi}_{sing.}$  column shows the average fertility of once-seen source words. For Czech-English, the year refers to the WMT shared task; for all other language pairs, the year refers to the NIST Open MT Evaluation. \*Half of this test set was also used for tuning feature weights.

- Czech-English: A corpus of 4 million words of Czech-English data from the News Commentary corpus.<sup>4</sup>

We set the hyperparameters  $\alpha$  and  $\beta$  by tuning on gold-standard word alignments (to maximize F1) when possible. For Arabic-English and Chinese-English, we used 346 and 184 hand-aligned sentences from LDC2006E86 and LDC2006E93. Similarly, for Czech-English, 515 hand-aligned sentences were available (Bojar and Prokopová, 2006). But for Urdu-English, since we did not have any gold alignments, we used  $\alpha = 10$  and  $\beta = 0.05$ . We did not choose a large  $\alpha$ , as the dataset was small, and we chose a conservative value for  $\beta$ .

We ran word alignment in both directions and symmetrized using *grow-diag-final* (Koehn et al., 2003). For models with the smoothed  $\ell_0$  prior, we tuned  $\alpha$  and  $\beta$  separately in each direction.

### 3.2 Alignment

First, we evaluated alignment accuracy directly by comparing against gold-standard word alignments.

<sup>4</sup>This data is available at <http://statmt.org/wmt10>.

The results are shown in the *alignment F1* column of Table 1. We used balanced F-measure rather than alignment error rate as our metric (Fraser and Marcu, 2007).

Following Dyer et al. (2011), we also measured the average fertility,  $\tilde{\phi}_{sing.}$ , of once-seen source words in the symmetrized alignments. Our alignments show smaller fertility for once-seen words, suggesting that they suffer from “garbage collection” effects less than the baseline alignments do.

The fact that we had to use hand-aligned data to tune the hyperparameters  $\alpha$  and  $\beta$  means that our method is no longer completely unsupervised. However, our observation is that alignment accuracy is actually fairly robust to the choice of these hyperparameters, as shown in Table 2. As we will see below, we still obtained strong improvements in translation quality when hand-aligned data was unavailable.

We also tried generating 50 word classes using the tool provided in GIZA++. We found that adding word classes improved alignment quality a little, but more so for the baseline system (see Table 3). We used the alignments generated by training with word classes for our translation experiments.



$\beta$	model	$\alpha$								
		0	10	25	50	75	100	250	500	750
-	HMM	47.5								
	M4	52.1								
0.5	HMM		46.3	48.4	52.8	55.7	57.5	61.5	62.6	<b>62.7</b>
	M4		51.7	53.7	56.4	58.6	59.8	63.3	64.4	64.8
0.1	HMM		55.6	60.4	61.6	62.1	61.9	61.8	60.2	60.1
	M4		58.2	62.4	64.0	64.4	64.8	65.5	65.6	<b>65.9</b>
0.05	HMM		59.1	61.4	62.4	62.5	62.3	60.8	58.7	57.7
	M4		61.0	63.5	64.6	65.3	65.3	65.4	65.7	65.7
0.01	HMM		59.7	61.6	60.0	59.5	58.7	56.9	55.7	54.7
	M4		62.9	65.0	65.1	65.2	65.1	65.4	65.3	65.4
0.005	HMM		58.1	59.0	58.3	57.6	57.0	55.9	53.9	51.7
	M4		62.0	64.1	64.5	64.5	64.5	65.0	64.8	64.6
0.001	HMM		51.7	52.1	51.4	49.3	50.4	46.8	45.4	44.0
	M4		59.8	61.3	61.5	61.0	61.8	61.2	61.0	61.2

Table 2: Almost all hyperparameter settings achieve higher F-scores than the baseline IBM Model 4 and HMM model for Arabic-English alignment ( $\alpha = 0$ ).

direction	system	word classes?	
		no	yes
$P(f e)$	baseline	49.0	52.1
	$\ell_0$ -norm	<b>63.9</b>	<b>65.9</b>
	difference	+14.9	+13.8
$P(e f)$	baseline	64.3	65.2
	$\ell_0$ -norm	<b>69.2</b>	<b>70.3</b>
	difference	+4.9	+5.1

Table 3: Adding word classes improves the F-score in both directions for Arabic-English alignment by a little, for the baseline system more so than ours.

Figure 2 shows four examples of Chinese-English alignment, comparing the baseline with our smoothed- $\ell_0$  method. In all four cases, the baseline produces incorrect extra alignments that prevent good translation rules from being extracted while the smoothed- $\ell_0$  results are correct. In particular, the baseline system demonstrates typical “garbage collection” behavior (Moore, 2004) in all four examples.

### 3.3 Translation

We then tested the effect of word alignments on translation quality using the hierarchical phrase-based translation system Hiero (Chiang, 2007). We used a fairly standard set of features: seven inherited from Pharaoh (Koehn et al., 2003), a sec-

setting		align F1 (%)	B (%)	
$t(f e)$	$t(e f)$		2008	2009
1st	1st	70.8	41.1	43.7
1st	2nd	70.7	41.1	43.8
2nd	1st	70.7	40.7	44.1
2nd	2nd	70.9	41.1	44.2

Table 4: Optimizing hyperparameters on alignment F1 score does not necessarily lead to optimal B . The first two columns indicate whether we used the first- or second-best alignments in each direction (according to F1); the third column shows the F1 of the symmetrized alignments, whose corresponding B scores are shown in the last two columns.

ond language model, and penalties for the glue rule, identity rules, unknown-word rules, and two kinds of number/name rules. The feature weights were discriminatively trained using MIRA (Chiang et al., 2008). We used two 5-gram language models, one on the combined English sides of the NIST 2009 Arabic-English and Chinese-English constrained tracks (385M words), and another on 2 billion words of English.

For each language pair, we extracted grammar rules from the same data that were used for word alignment. The development data that were used for discriminative training were: for Chinese-English and Arabic-English, data from the NIST 2004 and NIST 2006 test sets, plus newsgroup data from the

GALE program (LDC2006E92); for Urdu-English, half of the NIST 2008 test set; for Czech-English, a training set of 2051 sentences provided by the WMT10 translation workshop.

The results are shown in the B column of Table 1. We used case-insensitive IBM B (closest reference length) as our metric. Significance testing was carried out using bootstrap resampling with 1000 samples (Koehn, 2004; Zhang et al., 2004).

All of the tests showed significant improvements ( $p < 0.01$ ), ranging from +0.4 B to +1.4 B. For Urdu, even though we didn't have manual alignments to tune hyperparameters, we got significant gains over a good baseline. This is promising for languages that do not have any manually aligned data.

Ideally, one would want to tune  $\alpha$  and  $\beta$  to maximize B. However, this is prohibitively expensive, especially if we must tune them separately in each alignment direction before symmetrization. We ran some contrastive experiments to investigate the impact of hyperparameter tuning on translation quality. For the smaller Arabic-English corpus, we symmetrized all combinations of the two top-scoring alignments (according to F1) in each direction, yielding four sets of alignments. Table 4 shows B scores for translation models learned from these alignments. Unfortunately, we find that optimizing F1 is not optimal for B—using the second-best alignments yields a further improvement of 0.5 B on the NIST 2009 data, which is statistically significant ( $p < 0.05$ ).

## 4 Related Work

Schoenemann (2011a), taking inspiration from Bodrumlu et al. (2009), uses integer linear programming to optimize IBM Model 1–2 and the HMM with the  $\ell_0$ -norm. This method, however, does not outperform GIZA++. In later work, Schoenemann (2011b) used projected gradient descent for the  $\ell_1$ -norm. Here, we have adopted his use of projected gradient descent, but using a smoothed  $\ell_0$ -norm.

Liang et al. (2006) show how to train IBM models in both directions simultaneously by adding a term to the log-likelihood that measures the agreement between the two directions. Graça et al. (2010) explore modifications to the HMM model that encourage bijectivity and symmetry. The modifications

take the form of constraints on the posterior distribution over alignments that is computed during the E-step. Mermer and Saraçlar (2011) explore a Bayesian version of IBM Model 1, applying sparse Dirichlet priors to  $t$ . However, because this method requires the use of Monte Carlo methods, it is not clear how well it can scale to larger datasets.

## 5 Conclusion

We have extended the IBM models and HMM model by the addition of an  $\ell_0$  prior to the word-to-word translation model, which compacts the word-to-word translation table, reducing overfitting, and, in particular, the “garbage collection” effect. We have shown how to perform MAP-EM with this prior efficiently, even for large datasets. The method is implemented as a modification to the open-source toolkit GIZA++, and we have shown that it significantly improves translation quality across four different language pairs. Even though we have used a small set of gold-standard alignments to tune our hyperparameters, we found that performance was fairly robust to variation in the hyperparameters, and translation performance was good even when gold-standard alignments were unavailable. We hope that our method, due to its simplicity, generality, and effectiveness, will find wide application for training better statistical translation systems.

## Acknowledgments

We are indebted to Thomas Schoenemann for initial discussions and pilot experiments that led to this work, and to the anonymous reviewers for their valuable comments. We thank Jason Riesa for providing the Arabic-English and Chinese-English hand-aligned data and the alignment visualization tool, and Chris Dyer for the Czech-English hand-aligned data. This research was supported in part by DARPA under contract DOI-NBC D11AP00244 and a Google Faculty Research Award to L. H.

## References

- Andrew Barron, Jorma Rissanen, and Bin Yu. 1998. The minimum description length principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760.
- Dimitri P. Bertsekas. 1999. *Nonlinear Programming*. Athena Scientific.
- Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.
- Tugba Bodrumlu, Kevin Knight, and Sujith Ravi. 2009. A new objective function for word alignment. In *Proceedings of the NAACL HLT Workshop on Integer Linear Programming for Natural Language Processing*.
- Ondřej Bojar and Magdalena Prokopová. 2006. Czech-English word alignment. In *Proceedings of LREC*.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19:263–311.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of EMNLP*.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–208.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Computational Linguistics*, 39(4):1–38.
- John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. 2008. Efficient projections onto the  $\ell_1$ -ball for learning in high dimensions. In *Proceedings of ICML*.
- Chris Dyer, Jonathan H. Clark, Alon Lavie, and Noah A. Smith. 2011. Unsupervised word alignment with arbitrary features. In *Proceedings of ACL*.
- Alexander Fraser and Daniel Marcu. 2007. Measuring word alignment quality for statistical machine translation. *Computational Linguistics*, 33(3):293–303.
- João V. Graça, Kuzman Ganchev, and Ben Taskar. 2010. Learning tractable word alignment models with complex constraints. *Computational Linguistics*, 36(3):481–504.
- Philipp Koehn, Franz Joseph Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL*.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*.
- Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of HLT-NAACL*.
- Coşkun Mermer and Murat Saraçlar. 2011. Bayesian word alignment for statistical machine translation. In *Proceedings of ACL HLT*.
- Robert C. Moore. 2004. Improving IBM word-alignment Model 1. In *Proceedings of ACL*.
- Robert Moore. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of HLT-EMNLP*.
- Franz Joseph Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417–449.
- Jason Riesa and Daniel Marcu. 2010. Hierarchical search for word alignment. In *Proceedings of ACL*.
- Thomas Schoenemann. 2011a. Probabilistic word alignment under the  $L_0$ -norm. In *Proceedings of CoNLL*.
- Thomas Schoenemann. 2011b. Regularizing mono- and bi-word models for word alignment. In *Proceedings of IJCNLP*.
- Ben Taskar, Lacoste-Julien Simon, and Klein Dan. 2005. A discriminative matching approach to word alignment. In *Proceedings of HLT-EMNLP*.
- Ashish Vaswani, Adam Pauls, and David Chiang. 2010. Efficient optimization of an MDL-inspired objective function for unsupervised part-of-speech tagging. In *Proceedings of ACL*.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of COLING*.
- 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*.

# Modeling Review Comments

**Arjun Mukherjee**

Department of Computer Science  
University of Illinois at Chicago  
Chicago, IL 60607, USA  
arjun4787@gmail.com

**Bing Liu**

Department of Computer Science  
University of Illinois at Chicago  
Chicago, IL 60607, USA  
liub@cs.uic.edu

## Abstract

Writing comments about news articles, blogs, or reviews have become a popular activity in social media. In this paper, we analyze reader comments about reviews. Analyzing review comments is important because reviews only tell the experiences and evaluations of reviewers about the reviewed products or services. Comments, on the other hand, are readers' evaluations of reviews, their questions and concerns. Clearly, the information in comments is valuable for both future readers and brands. This paper proposes two latent variable models to simultaneously model and extract these key pieces of information. The results also enable classification of comments accurately. Experiments using Amazon review comments demonstrate the effectiveness of the proposed models.

## 1. Introduction

Online reviews enable consumers to evaluate the products and services that they have used. These reviews are also used by other consumers and businesses as a valuable source of opinions.

However, reviews only give the evaluations and experiences of the reviewers. Often a reviewer may not be an expert of the product and may misuse the product or make other mistakes. There may also be aspects of the product that the reviewer did not mention but a reader wants to know. Some reviewers may even write fake reviews to promote

some products, which is called *opinion spamming* (Jindal and Liu 2008). To improve the online review system and user experience, some review hosting sites allow readers to write comments about reviews (apart from just providing a feedback by clicking whether the review is helpful or not). Many reviews receive a large number of comments. It is difficult for a reader to read them to get a gist of them. An automated comment analysis would be very helpful. Review comments mainly contain the following information:

**Thumbs-up or thumbs-down:** Some readers may comment on whether they find the review useful in helping them make a buying decision.

**Agreement or disagreement:** Some readers who comment on a review may be users of the product themselves. They often state whether they agree or disagree with the review. Such comments are valuable as they provide a second opinion, which may even identify fake reviews because a genuine user often can easily spot reviewers who have never used the product.

**Question and answer:** A commenter may ask for clarification or about some aspects of the product that are not covered in the review.

In this paper, we use statistical modeling to model review comments. Two new generative models are proposed. The first model is called the Topic and Multi-Expression model (TME). It models topics and different types of expressions, which represent different types of comment posts:

1. *Thumbs-up* (e.g., “review helped me”)
2. *Thumbs-down* (e.g., “poor review”)
3. *Question* (e.g., “how to”)

4. *Answer acknowledgement* (e.g., “thank you for clarifying”). Note that we have no expressions for answers to questions as there are usually no specific phrases indicating that a post answers a question except starting with the name of the person who asked the question. However, there are typical phrases for acknowledging answers, thus *answer acknowledgement* expressions.
5. *Disagreement (contention)* (e.g., “I disagree”)
6. *Agreement* (e.g., “I agree”).

For ease of presentation, we call these expressions the *comment expressions* (or *C-expressions*). TME provides a basic model for extracting these pieces of information and topics. Its generative process separates topics and C-expression types using a switch variable and treats posts as random mixtures over latent topics and C-expression types. The second model, called ME-TME, improves TME by using Maximum-Entropy priors to guide topic/expression switching. In short, the two models provide a principled and integrated approach to simultaneously discover topics and C-expressions, which is the goal of this work. Note that topics are usually product aspects in this work.

The extracted C-expressions and topics from review comments are very useful in practice. First of all, C-expressions enable us to perform more accurate classification of comments, which can give us a good evaluation of the review quality and credibility. For example, a review with many *Disagreeing* and *Thumbs-down* comments is dubious. Second, the extracted C-expressions and topics help identify the key product aspects that people are troubled with in disagreements and in questions. Our experimental results in Section 5 will demonstrate these capabilities of our models.

With these pieces of information, comments for a review can be summarized. The summary may include, but not limited to, the following: (1) percent of people who give the review thumbs-up or thumbs-down; (2) percent of people who agree or disagree (or contend) with the reviewer; (3) contentious (disagreed) aspects (or topics); (4) aspects about which people often have questions.

To the best of our knowledge, there is no reported work on such a fine-grained modeling of review comments. The related works are mainly in sentiment analysis (Pang and Lee, 2008; Liu 2012), e.g., topic and sentiment modeling, review quality prediction and review spam detection. However, our work is different from them. We will compare with them in detail in Section 2.

The proposed models have been evaluated both qualitatively and quantitatively using a large number of review comments from Amazon.com. Experimental results show that both TME and ME-TME are effective in performing their tasks. ME-TME also outperforms TME significantly.

## 2. Related Work

We believe that this work is the first attempt to model review comments for fine-grained analysis. There are, however, several general research areas that are related to our work.

Topic models such as LDA (Latent Dirichlet Allocation) (Blei et al., 2003) have been used to mine topics in large text collections. There have been various extensions to multi-grain (Titov and McDonald, 2008a), labeled (Ramage et al., 2009), partially-labeled (Ramage et al., 2011), constrained (Andrzejewski et al., 2009) models, etc. These models produce only topics but not multiple types of expressions together with topics. Note that in labeled models, each document is labeled with one or multiple labels. For our work, there is no label for each comment. Our labeling is on topical terms and C-expressions with the purpose of obtaining some priors to separate topics and C-expressions.

In sentiment analysis, researchers have jointly modeled topics and sentiment words (Lin and He, 2009; Mei et al., 2007; Lu and Zhai, 2008; Titov and McDonald, 2008b; Lu et al., 2009; Brody and Elhadad, 2010; Wang et al., 2010; Jo and Oh, 2011; Maghaddam and Ester, 2011; Sauper et al., 2011; Mukherjee and Liu, 2012a). Our model is more related to the ME-LDA model in (Zhao et al., 2010), which used a switch variable trained with Maximum-Entropy to separate topic and sentiment words. We also use such a variable. However, unlike sentiments and topics in reviews, which are emitted in the same sentence, C-expressions often interleave with topics across sentences and the same comment post may also have multiple types of C-expressions. Additionally, C-expressions are mostly phrases rather than individual words. Thus, a different model is required to model them.

There have also been works aimed at putting authors in debate into support/oppose camps, e.g., (Galley et al., 2004; Agarwal et al., 2003; Murakami and Raymond, 2010), modeling debate discussions considering reply relations (Mukherjee and Liu, 2012b), and identifying stances in debates (Somasundaran and Wiebe, 2009; Thomas et al.,

2006; Burfoot et al., 2011). (Yano and Smith, 2010) also modeled the relationship of a blog post and the number of comments it receives. These works are different as they do not mine C-expressions or discover the points of contention and questions in comments.

In (Kim et al., 2006; Zhang and Varadarajan, 2006; Ghose and Ipeirotis, 2007; Liu et al., 2007; Liu et al., 2008; O’Mahony and Smyth, 2009; Tsur and Rappoport 2009), various classification and regression approaches were taken to assess the quality of reviews. (Jindal and Liu, 2008; Lim et al., 2010; Li et al. 2011; Ott et al., 2011; Mukherjee et al., 2012) detect fake reviews and reviewers. However, all these works are not concerned with review comments.

### 3. The Basic TME Model

This section discusses TME. The next section discusses ME-TME, which improves TME. These models belong to the family of generative models for text where words and phrases ( $n$ -grams) are viewed as random variables, and a document is viewed as a bag of  $n$ -grams and each  $n$ -gram takes a value from a predefined vocabulary. In this work, we use up to 4-grams, i.e.,  $n = 1, 2, 3, 4$ . For simplicity, we use *terms* to denote both *words* (unigrams or 1-grams) and *phrases* ( $n$ -grams). We denote the entries in our vocabulary by  $v_{1...V}$  where  $V$  is the number of unique terms in the vocabulary. The entire corpus contains  $d_{1...D}$  documents. A document (e.g., comment post)  $d$  is represented as a vector of terms  $\mathbf{w}_d$  with  $N_d$  entries.  $W$  is the set of all observed terms with cardinality,  $|W| = \sum_d N_d$ .

The TME (Topic and Multi-Expression) model is a hierarchical generative model motivated by the joint occurrence of various types of expressions indicating *Thumbs-up*, *Thumbs-down*, *Question*, *Answer acknowledgement*, *Agreement*, and *Disagreement* and topics in comment posts. As before, these expressions are collectively called C-expressions. A typical comment post mentions a few topics (using semantically related topical terms) and expresses some viewpoints with one or more C-expression types (using semantically related expressions). This observation motivates the generative process of our model where documents (posts) are represented as random mixtures of latent topics and C-expression types. Each topic or C-expression type is characterized by a distribution over terms (words/phrases). Assume

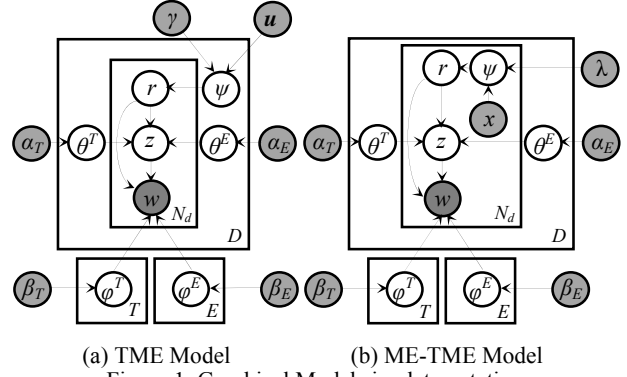


Figure 1: Graphical Models in plate notations.

we have  $t_{1...T}$  topics and  $e_{1...E}$  expression types in our corpus. Note that in our case of Amazon review comments, based on reading various posts, we hypothesize that  $E = 6$  as in such review discussions, we mostly find 6 expression types (more details in Section 5.1). Let  $\psi_d$  denote the distribution of topics and C-expressions in a document  $d$  with  $r_{d,j} \in \{\hat{t}, \hat{e}\}$  denoting the binary indicator variable (topic or C-expression) for the  $j^{th}$  term of  $d$ ,  $w_{d,j}$ .  $z_{d,j}$  denotes the appropriate topic or C-expression type index to which  $w_{d,j}$  belongs. We parameterize multinomials over topics using a matrix  $\Theta_{D \times T}^T$  whose elements  $\theta_{d,t}^T$  signify the probability of document  $d$  exhibiting topic  $t$ . For simplicity of notation, we will drop the latter subscript ( $t$  in this case) when convenient and use  $\theta_d^T$  to stand for the  $d^{th}$  row of  $\Theta^T$ . Similarly, we define multinomials over C-expression types using a matrix  $\Theta_{D \times E}^E$ . The multinomials over terms associated with each topic are parameterized by a matrix  $\Phi_{T \times V}^T$ , whose elements  $\phi_{t,v}^T$  denote the probability of generating  $v$  from topic  $t$ . Likewise, multinomials over terms associated with each C-expression type are parameterized by a matrix  $\Phi_{E \times V}^E$ . We now define the generative process of TME (see Figure 1(a)).

- A. For each C-expression type  $e$ , draw  $\varphi_e^E \sim \text{Dir}(\beta_E)$
- B. For each topic  $t$ , draw  $\varphi_t^T \sim \text{Dir}(\beta_T)$
- C. For each comment post  $d \in \{1 \dots D\}$ :
  - i. Draw  $\psi_d \sim \text{Beta}(\gamma \mathbf{u})$
  - ii. Draw  $\theta_d^E \sim \text{Dir}(\alpha_E)$
  - iii. Draw  $\theta_d^T \sim \text{Dir}(\alpha_T)$
  - iv. For each term  $w_{d,j}$ ,  $j \in \{1 \dots N_d\}$ :
    - a. Draw  $r_{d,j} \sim \text{Bernoulli}(\psi_d)$
    - b. if  $(r_{d,j} = \hat{e} // w_{d,j}$  is a C-expression term  
Draw  $z_{d,j} \sim \text{Mult}(\theta_d^E)$   
else  $// r_{d,j} = \hat{t}$ ,  $w_{d,j}$  is a topical term  
Draw  $z_{d,j} \sim \text{Mult}(\theta_d^T)$
    - c. Emit  $w_{d,j} \sim \text{Mult}(\varphi_{z_{d,j}}^{r_{d,j}})$

To learn the TME model from data, as exact inference is not possible, we resort to approximate inference using *collapsed Gibbs sampling* (Griffiths and Steyvers, 2004). Gibbs sampling is a form of Markov Chain Monte Carlo method where a Markov chain is constructed to have a particular stationary distribution. In our case, we want to construct a Markov chain which converges to the posterior distribution over  $R$  and  $Z$  conditioned on the data. We only need to sample  $z$  and  $r$  as we use collapsed Gibbs sampling and the dependencies of  $\theta$  and  $\varphi$  have been integrated out analytically in the joint. Denoting the random variables  $\{w, z, r\}$  by singular subscripts  $\{w_k, z_k, r_k\}$ ,  $k_{1\dots K}$ , where  $K = \sum_d N_d$ , a single iteration consists of performing the following sampling:

$$p(z_k = t, r_k = \hat{t} | W_{-k}, Z_{-k}, R_{-k}, w_k = v) \propto \frac{n_{d,-k}^T + \gamma_a}{n_{d,-k}^T + n_{d,-k}^E + \gamma_a + \gamma_b} \times \frac{n_{d,t,-k}^{DT} + \alpha_T}{n_{d,(\cdot),-k}^{DT} + T\alpha_T} \times \frac{n_{t,v,-k}^{CT} + \beta_T}{n_{t,(\cdot),-k}^{CT} + V\beta_T} \quad (1)$$

$$p(z_k = e, r_k = \hat{e} | W_{-k}, Z_{-k}, R_{-k}, w_k = v) \propto \frac{n_{d,-k}^E + \gamma_b}{n_{d,-k}^T + n_{d,-k}^E + \gamma_a + \gamma_b} \times \frac{n_{d,e,-k}^{DE} + \alpha_E}{n_{d,(\cdot),-k}^{DE} + E\alpha_E} \times \frac{n_{e,v,-k}^{CE} + \beta_E}{n_{e,(\cdot),-k}^{CE} + V\beta_E} \quad (2)$$

where  $k = (d, j)$  denotes the  $j^{\text{th}}$  term of document  $d$  and the subscript  $-k$  denotes assignments excluding the term at  $(d, j)$ . Counts  $n_{t,v}^{CT}$  and  $n_{e,v}^{CE}$  denote the number of times term  $v$  was assigned to topic  $t$  and expression type  $e$  respectively.  $n_{d,t}^{DT}$  and  $n_{d,e}^{DE}$  denote the number of terms in document  $d$  that were assigned to topic  $t$  and C-expression type  $e$  respectively. Lastly,  $n_d^T$  and  $n_d^E$  are the number of terms in  $d$  that were assigned to topics and C-expression types respectively. Omission of the latter index denoted by  $(\cdot)$  represents the marginalized sum over the latter index. We employ a blocked sampler jointly sampling  $r$  and  $z$  as this improves convergence and reduces autocorrelation of the Gibbs sampler (Rosen-Zvi et al., 2004).

**Asymmetric Beta priors:** Based on our initial experiments with TME, we found that properly setting the smoothing hyper-parameter  $\gamma \mathbf{u}$  is crucial as it governs the topic/expression switch.

According to the generative process,  $\psi_d$  is the (success) probability (of the Bernoulli distribution) of emitting a topical/aspect term in a comment post  $d$  and  $1 - \psi_d$ , the probability of emitting a C-expression term in  $d$ . Without loss of generality, we draw  $\psi_d \sim \text{Beta}(\gamma \mathbf{u})$  where  $\gamma$  is the concentration parameter and  $\mathbf{u} = [u_a, u_b]$  is the base measure. Without any prior belief, one resorts

to uniform base measure  $u_a = u_b = 0.5$  (i.e., assumes that both topical and C-expression terms are equally likely to be emitted in a comment post). This results in symmetric Beta priors  $\psi_d \sim \text{Beta}(\gamma_a, \gamma_b)$  where  $\gamma_a = \gamma u_a$ ,  $\gamma_b = \gamma u_b$  and  $\gamma_a = \gamma_b = \gamma/2$ . However, knowing the fact that topics are more likely to be emitted than expressions in a post *a priori* motivates us to take guidance from asymmetric priors (i.e., we now have a non-uniform base measure  $\mathbf{u}$ ). This asymmetric setting of  $\gamma$  ensures that samples of  $\psi_d$  are more close to the actual distribution of topical terms in posts based on some domain knowledge. Symmetric  $\gamma$  cannot utilize any prior knowledge. In (Lin and He, 2009), a method was proposed to incorporate domain knowledge during Gibbs sampling initialization, but its effect becomes weak as the sampling progresses (Jo and Oh, 2011).

For asymmetric priors, we estimate the hyper-parameters from labeled data. Given a labeled set  $D_L$ , where we know the per post probability of C-expression emission ( $1 - \psi_d$ ), we use the method of moments to estimate  $\gamma = [\gamma_a, \gamma_b]$  as follows:

$$\gamma_a = \mu \left( \frac{\mu(1-\mu)}{\sigma} - 1 \right), \gamma_b = \gamma_a \left( \frac{1}{\mu} - 1 \right); \mu = E[\psi_d], \sigma = \text{Var}[\psi_d] \quad (3)$$

#### 4. ME-TME Model

The guidance of Beta priors, although helps, is still relatively coarse and weak. We can do better to produce clearer separation of topical and C-expression terms. An alternative strategy is to employ *Maximum-Entropy* (Max-Ent) priors instead of Beta priors. The Max-Ent parameters can be learned from a small number of labeled topical and C-expression terms (words and phrases) which can serve as good priors. The idea is motivated by the following observation: topical and C-expression terms typically play different syntactic roles in a sentence. Topical terms (e.g. “ipod” “cell phone”, “macro lens”, “kindle”, etc.) tend to be noun and noun phrases while expression terms (“I refute”, “how can you say”, “great review”) usually contain pronouns, verbs, wh-determiners, adjectives, and modals. In order to utilize the part-of-speech (POS) tag information, we move the topic/C-expression distribution  $\psi_d$  (the prior over the indicator variable  $r_{d,j}$ ) from the document plate to the word plate (see Figure 1 (b)) and draw it from a Max-Ent model conditioned on the observed feature vector  $\vec{x}_{d,j}$  associated with  $w_{d,j}$  and the learned Max-Ent parameters  $\lambda$ .  $x_{d,j}$  can

encode arbitrary contextual features for learning. With Max-Ent priors, we have the new model ME-TME. In this work, we encode both lexical and POS features of the previous, current and next POS tags/lexemes of the term  $w_{d,j}$ . More specifically,

$$\vec{x}_{d,j} = [POS_{w_{d,j-1}}, POS_{w_{d,j}}, POS_{w_{d,j+1}}, w_{d,j} - 1, w_{d,j}, w_{d,j} + 1]$$

For phrasal terms (n-grams), all POS tags and lexemes of  $w_{d,j}$  are considered as features. Incorporating Max-Ent priors, the Gibbs sampler of ME-TME is given by:

$$p(z_k = t, r_k = \hat{t} | W_{-k}, Z_{-k}, R_{-k}, w_k = v) \propto \frac{\exp(\sum_{i=1}^n \lambda_i f_i(x_{d,j}, \hat{t}))}{\sum_{y \in \{\hat{e}, \hat{t}\}} \exp(\sum_{i=1}^n \lambda_i f_i(x_{d,j}, y))} \times \frac{n_{d,t}^{DT} + \alpha_T}{n_{d,(c)}^{DT} + T \alpha_T} \times \frac{n_{\hat{t},v}^{CT} + \beta_T}{n_{\hat{t},(c)}^{CT} + V \beta_T} \quad (4)$$

$$p(z_k = e, r_k = \hat{e} | W_{-k}, Z_{-k}, R_{-k}, w_k = v) \propto \frac{\exp(\sum_{i=1}^n \lambda_i f_i(x_{d,j}, \hat{e}))}{\sum_{y \in \{\hat{e}, \hat{t}\}} \exp(\sum_{i=1}^n \lambda_i f_i(x_{d,j}, y))} \times \frac{n_{d,e}^{DE} + \alpha_E}{n_{d,(c)}^{DE} + E \alpha_E} \times \frac{n_{\hat{e},v}^{CE} + \beta_E}{n_{\hat{e},(c)}^{CE} + V \beta_E} \quad (5)$$

where  $\lambda_{1\dots n}$  are the parameters of the learned Max-Ent model corresponding to the  $n$  binary feature functions  $f_{1\dots n}$  from Max-Ent.

## 5. Evaluation

We now evaluate the proposed TME and ME-TME models. Specifically, we evaluate the discovered C-expressions, contentious aspects, and aspects often mentioned in questions.

**5.1 Dataset and Experiment Settings** We crawled comments of reviews in Amazon.com for a variety of products. For each comment we extracted its id, the comment author id, the review id on which it commented, and the review author id. Our database consisted of 21,316 authors, 37,548 reviews, and 88,345 comments with an average of 124 words per comment post.

For all our experiments, the hyper-parameters for TME and ME-TME were set to the heuristic values  $\alpha_T = 50/T$ ,  $\alpha_E = 50/E$ ,  $\beta_T = \beta_E = 0.1$  as suggested in (Griffiths and Steyvers, 2004). For  $\gamma$ , we estimated the asymmetric Beta priors using the method of moments discussed in Section 3. We sampled 1000 random posts and for each post we identified the C-expressions emitted. We thus computed the per-post probability of C-expression emission ( $1 - \psi_d$ ) and used Eq. (3) to get the final estimates,  $\gamma_a = 3.66$ ,  $\gamma_b = 1.21$ . To learn the Max-Ent parameters  $\lambda$ , we randomly sampled 500 terms from our corpus appearing at least 10 times and labeled them as topical (332) or C-expressions (168) and used the corresponding feature vector of

each term (in the context of posts where it occurs) to train the Max-Ent model. We set the number of topics,  $T = 100$  and the number of C-expression types,  $E = 6$  (*Thumbs-up*, *Thumbs-down*, *Question*, *Answer acknowledgement*, *Agreement* and *Disagreement*) as in review comments, we usually find these six dominant expression types. Note that knowing the exact number of topics,  $T$  and expression types,  $E$  in a corpus is difficult. While non-parametric Bayesian approaches (Teh et al., 2006) aim to estimate  $T$  from the corpus, in this work the heuristic values obtained from our initial experiments produced good results. We also tried increasing  $E$  to 7, 8, etc. However, it did not produce any new dominant expression type. Instead, the expression types became less specific as the expression term space became sparser.

## 5.2 C-Expression Evaluation

We now evaluate the discovered C-expressions. We first evaluate them qualitatively in Tables 1 and 2. Table 1 shows the top terms of all expression types using the TME model. We find that TME can discover and cluster many correct C-expressions, e.g., “great review”, “review helped me” in *Thumbs-up*; “poor review”, “very unfair review” in *Thumbs-down*; “how do I”, “help me decide” in *Question*; “good reply”, “thank you for clarifying” in *Answer Acknowledgement*; “I disagree”, “I refute” in *Disagreement*; and “I agree”, “true in fact” in *Agreement*. However, with the guidance of Max-Ent priors, ME-TME did much better (Table 2). For example, we find “level headed review”, “review convinced me” in *Thumbs-up*; “biased review”, “is flawed” in *Thumbs-down*; “any clues”, “I was wondering how” in *Question*; “clears my”, “valid answer” in *Answer-acknowledgement*; “I don’t buy your”, “sheer nonsense” in *Disagreement*; “agree completely”, “well said” in *Agreement*. These newly discovered phrases by ME-TME are marked in *blue* in Table 3. ME-TME also has fewer errors.

Next, we evaluate them quantitatively using the metric *precision @ n*, which gives the precision at different rank positions. This metric is appropriate here because the C-expressions (according to top terms in  $\Phi^E$ ) produced by TME and ME-TME are rankings. Table 3 reports the precisions @ top 25, 50, 75, and 100 rank positions for all six expression types across both models. We evaluated till top 100 positions because it is usually



**Thumbs-up (e<sub>1</sub>):** **review, thanks**, great review, nice review, **time**, best review, **appreciate, you**, your review helped, nice, terrific, review helped me, good critique, **very, assert, wrong**, useful review, **don't, misleading**, thanks a lot, ...

**Thumbs-down (e<sub>2</sub>):** **review, no**, poor review, imprecise, **you, complaint, very**, suspicious, bogus review, **absolutely, credible**, very unfair review, criticisms, **true**, disregard this review, disagree with, **judgment**, without owning, ...

**Question (e<sub>3</sub>):** question, **my, I**, how do I, why isn't, please explain, **good answer**, clarify, don't understand, my doubts, I'm confused, **does not, understand**, help me decide, how to, **yes, answer**, how can I, **can't explain**, ...

**Answer Acknowledgement (e<sub>4</sub>):** **my, informative**, answer, good reply, thank you for clarifying, answer doesn't, good answer, **vague**, helped me choose, useful suggestion, **don't understand, cannot explain**, your answer, **doubts**, answer isn't, ...

**Disagreement (e<sub>5</sub>):** disagree, **I, don't**, I disagree, argument claim, I reject, I refute, I refuse, oppose, debate, **accept**, don't agree, **quote, sense**, would disagree, **assertions**, I doubt, **right**, your, **really**, you, I'd disagree, cannot, nonsense, ...

**Agreement (e<sub>6</sub>):** yes, **do**, correct, indeed, **no**, right, I agree, **you**, agree, I accept, **very**, yes indeed, true in fact, indeed correct, I'd agree, **completely**, true, **but, doesn't, don't**, definitely, **false**, completely agree, agree with your, true, ...

Table 1: Top terms (comma delimited) of six expression types e<sub>1</sub>, e<sub>2</sub>, e<sub>3</sub>, e<sub>4</sub>, e<sub>5</sub>, e<sub>6</sub> ( $\Phi^6$ ) using TME model. **Red (bold)** colored terms denote possible errors

important to see whether a model can discover and rank those major expressions of a type at the top. We believe that top 100 are sufficient for most applications. From Table 3, we observe that ME-TME consistently outperforms TME in precisions across all expression types and all rank positions. This shows that Max-Ent priors are more effective in discovering expressions than Beta priors. Note that we couldn't compare with an existing baseline because there is no reported study on this problem.

### 5.3 Comment Classification

Here we show that the discovered C-expressions can help comment classification. Note that since a comment can belong to one or more types (e.g., a comment can belong to both *Thumbs-up* and *Agreement* types), this task is an instance of multi-label classification, i.e., an instance can have more than one class label. In order to evaluate all the expression types, we follow the binary approach which is an extension of one-against-all method for multi-label classification. Thus, for each label, we build a binary classification problem. Instances associated with that label are in one class and the rest are in the other class. To perform this task, we randomly sampled 2000 comments, and labeled each of them into one or more of the following 8 labels: *Thumbs-up*, *Thumbs-down*, *Disagreement*, *Agreement*, *Question*, *Answer-Acknowledgement*, *Answer*, and *None*, which have 432, 401, 309, 276,

**Thumbs-up (e<sub>1</sub>):** **review, you**, great review, *I'm glad I read*, best review, *review convinced me*, review helped me, good review, *terrific review, job, thoughtful review*, awesome review, *level headed review, good critique*, good job, *video review*, ...

**Thumbs-down (e<sub>2</sub>):** **review, you**, bogus review, con, useless review, *ridiculous, biased review*, very unfair review, *is flawed, completely, skeptical, badmouth, misleading review*, cynical review, wrong, disregard this review, *seemingly honest*, ...

**Question (e<sub>3</sub>):** question, **I**, how do I, why isn't, please explain, clarify, *any clues, answer*, please explain, help me decide, **vague**, how to, how do I, where can I, *how to set, I was wondering how, could you explain*, how can I, *can I use*, ...

**Answer Acknowledgement (e<sub>4</sub>):** **my**, good reply, , answer, reply, helped me choose, *clears my, valid answer*, answer doesn't, *satisfactory answer, can you clarify, informative answer*, useful suggestion, *perfect answer*, thanks for your reply, **doubts**, ...

**Disagreement (e<sub>5</sub>):** disagree, **I, don't**, I disagree, doesn't, *I don't buy your, credible*, I reject, I doubt, I refuse, *I oppose, sheer nonsense, hardly*, don't agree, *can you prove, you have no clue, how do you say, sense, you fail, contradiction*, ...

**Agreement (e<sub>6</sub>):** **I, do**, agree, **point**, yes, **really, would agree, you**, agree, I accept, **claim, agree completely, personally agree**, true in fact, indeed correct, *well said, valid point*, correct, **never meant, might not, definitely agree**, ...

Table 2: Top terms (comma delimited) of six expression types using ME-TME model. **Red (bold)** terms denote possible errors. *Blue (italics)* terms denote those newly discovered by the model; rest (black) were used in Max-Ent training.

305, 201, 228, and 18 comments respectively. We disregard the *None* category due to its small size. This labeling is a fairly easy task as one can almost certainly make out to which type a comment belongs. Thus we didn't use multiple labelers. The distribution reveals that the labels are overlapping. For instance, we found many comments belonging to both *Thumbs-down* and *Disagreement*, *Thumbs-up* with *Acknowledgement* and with *Question*.

For supervised classification, the choice of feature is a key issue. While word and POS n-grams are traditional features, such features may not be the best for our task. We now compare such features with the C-expressions discovered by the proposed models. We used the top 1000 terms from each of the 6 C-expression rankings as features. As comments in *Question* type mostly use the punctuation "?", we added it in our feature set. We use precision, recall and F<sub>1</sub> as our metric to compare classification performance using a trained SVM (linear kernel). All results (Table 4) were computed using 10-fold cross-validation (CV). We also tried Naïve Bayes and Logistic Regression classifiers, but they were poorer than SVM. Hence their results are not reported due to space constraints. As a separate experiment (not shown here also due to space constraints), we analyzed the classification performance by varying the number of top terms from 200, 400, ..., 1000, 1200, etc. and found that the F<sub>1</sub> scores stabilized after top

C-Expression Type	P@25		P@50		P@75		P@100	
	TME	ME-TME	TME	ME-TME	TME	ME-TME	TME	ME-TME
Thumbs-up	0.60	0.80	0.66	0.78	0.60	0.69	0.55	0.64
Thumbs-down	0.68	0.84	0.70	0.80	0.63	0.67	0.60	0.65
Question	0.64	0.80	0.68	0.76	0.65	0.72	0.61	0.67
Answer-Acknowledgement	0.68	0.76	0.62	0.72	0.57	0.64	0.54	0.58
Disagreement	0.76	0.88	0.74	0.80	0.68	0.73	0.65	0.70
Agreement	0.72	0.80	0.64	0.74	0.61	0.70	0.60	0.69

Table 3: Precision @ top 25, 50, 75, and 100 rank positions for all C-expression types.

Features	Thumbs-up			Thumbs-down			Question			Answer-Ack.			Disagreement			Agreement			Answer		
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>
W+POS 1-gram	0.68	0.66	0.67	0.65	0.65	0.65	0.71	0.68	0.69	0.64	0.61	0.62	0.73	0.72	0.72	0.67	0.65	0.66	0.58	0.57	0.57
W+POS 1-2 gram	0.72	0.69	0.70	0.68	0.67	0.67	0.74	0.69	0.71	0.69	0.63	0.65	0.76	0.75	0.75	0.71	0.69	0.70	0.60	0.57	0.58
W+POS, 1-3 gram	0.73	0.71	0.72	0.69	0.68	0.68	0.75	0.69	0.72	0.70	0.64	0.66	0.76	0.76	0.76	0.72	0.70	0.71	0.61	0.58	0.59
W+POS, 1-4 gram	0.74	0.72	0.73	0.71	0.68	0.69	0.75	0.70	0.72	0.70	0.65	0.67	0.77	0.76	0.76	0.73	0.70	0.71	0.61	0.58	0.59
C-Expr. $\Phi^E$ , TME	0.82	0.74	0.78	0.77	0.71	0.74	0.83	0.75	0.78	0.75	0.72	0.73	0.83	0.80	0.81	0.78	0.75	0.76	0.66	0.61	0.63
C-Expr. $\Phi^E$ , ME-TME	0.87	0.79	0.83	0.80	0.73	0.76	0.87	0.76	0.81	0.77	0.72	0.74	0.86	0.81	0.83	0.81	0.77	0.79	0.67	0.61	0.64

Table 4: Precision (P), Recall (R), and F<sub>1</sub> scores of binary classification using SVM and different features. The improvements of our models are significant ( $p < 0.001$ ) over paired  $t$ -test across 10-fold cross validation.

D	$\Phi^E$ + Noun/Noun Phrase						TME						ME-TME					
	J <sub>1</sub>			J <sub>2</sub>			J <sub>1</sub>			J <sub>2</sub>			J <sub>1</sub>			J <sub>2</sub>		
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>
D1	0.62	0.70	0.66	0.58	0.67	0.62	0.66	0.75	0.70	0.62	0.70	0.66	0.67	0.79	0.73	0.64	0.74	0.69
D2	0.61	0.67	0.64	0.57	0.63	0.60	0.66	0.72	0.69	0.62	0.67	0.64	0.68	0.75	0.71	0.64	0.71	0.67
D3	0.60	0.69	0.64	0.56	0.64	0.60	0.64	0.73	0.68	0.60	0.67	0.63	0.67	0.76	0.71	0.63	0.72	0.67
D4	0.59	0.68	0.63	0.55	0.65	0.60	0.63	0.71	0.67	0.59	0.68	0.63	0.65	0.73	0.69	0.62	0.71	0.66
Avg.	0.61	0.69	0.64	0.57	0.65	0.61	0.65	0.73	0.69	0.61	0.68	0.64	0.67	0.76	0.71	0.63	0.72	0.67

Table 5 (a)

D	$\Phi^E$ + Noun/Noun Phrase						TME						ME-TME					
	J <sub>1</sub>			J <sub>2</sub>			J <sub>1</sub>			J <sub>2</sub>			J <sub>1</sub>			J <sub>2</sub>		
	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>	P	R	F <sub>1</sub>
D1	0.57	0.65	0.61	0.54	0.63	0.58	0.61	0.69	0.65	0.58	0.66	0.62	0.64	0.73	0.68	0.61	0.70	0.65
D2	0.61	0.66	0.63	0.58	0.61	0.59	0.64	0.68	0.66	0.60	0.64	0.62	0.68	0.70	0.69	0.65	0.69	0.67
D3	0.60	0.68	0.64	0.57	0.64	0.60	0.64	0.71	0.67	0.62	0.68	0.65	0.67	0.72	0.69	0.64	0.69	0.66
D4	0.56	0.67	0.61	0.55	0.65	0.60	0.60	0.72	0.65	0.58	0.68	0.63	0.63	0.75	0.68	0.61	0.71	0.66
Avg.	0.59	0.67	0.62	0.56	0.63	0.59	0.62	0.70	0.66	0.60	0.67	0.63	0.66	0.73	0.69	0.63	0.70	0.66

Table 5 (b)

Table 5: Points of Contention (a), Questioned aspects (b). D1: Ipod, D2: Kindle, D3: Nikon, D4: Garmin. We report the average precision (P), recall (R), and F<sub>1</sub> score over 100 comments for each particular domain.

Statistical significance: Differences between Nearest Noun Phrase and TME for both judges (J<sub>1</sub>, J<sub>2</sub>) across all domains were significant at 97% confidence level ( $p < 0.03$ ). Differences among TME and ME-TME for both judges (J<sub>1</sub>, J<sub>2</sub>) across all domains were significant at 95% confidence level ( $p < 0.05$ ). A paired  $t$ -test was used for testing significance.

1000 terms. From Table 4, we see that F<sub>1</sub> scores dramatically increase with C-expression ( $\Phi^E$ ) features for all expression types. TME and ME-TME progressively improve the classification. Improvements of TME and ME-TME being significant ( $p < 0.001$ ) using a paired  $t$ -test across 10-fold cross validations shows that the discovered C-expressions are of high quality and useful.

We note that the annotation resulted in a new label “Answer” which consists of mostly replies to comments with questions. Since an “answer” to a question usually does not show any specific expression, it does not attain very good F<sub>1</sub> scores. Thus, to improve the performance of the *Answer*

type comments, we added three binary features for each comment  $c$  on top of C-expression features:

- i) Is the author of  $c$  the review author too? The idea here is that most of the times the reviewer answers the questions raised in comments.
- ii) Is there any comment posted before  $c$  by some author  $a$  which has been previously classified as a question post?
- iii) Is there any comment posted after  $c$  by author  $a$  that replies to  $c$  (using @name) and is an *Answer-Acknowledgement* comment (which again has been previously classified as such)?

Using these additional features, we obtained a precision of 0.78 and a recall of 0.73 yielding an F<sub>1</sub>

score of 0.75 which is a dramatic increase beyond 0.64 achieved by ME-TME in Table 4.

#### 5.4 Contention Points and Questioned Aspects

We now turn to the task of discovering points of contention in disagreement comments and aspects (or topics) raised in questions. By “points”, we mean the topical terms on which some contentions or disagreements have been expressed. Topics being the product aspects are also indirectly evaluated in this task. We employ the TME and ME-TME models in the following manner.

We only detail the approach for disagreement comments. The same method is applied to question comments. Given a disagreement comment post  $d$ , we first select the top  $k$  topics that are mentioned in  $d$  according to its topic distribution,  $\theta_d^T$ . Let  $T_d$  be the set of these top  $k$  topics in  $d$ . Then, for each disagreement expression  $w \in d \cap \varphi_{e=Disagreement}^E$ , we emit the topical terms (words/phrases) of topics in  $T_d$  which appear within a word window of  $q$  from  $w$  in  $d$ . More precisely, we emit the set  $A = \{w | w \in d \cap \varphi_t^T, t \in T_d, |posi(w) - posi(v)| \leq q\}$ , where  $posi(\cdot)$  returns the position index of the word or phrase in document  $d$ . To compute the intersection  $w \in d \cap \varphi_t^T$ , we need a threshold. This is so because the Dirichlet distribution has a smoothing effect which assigns some non-zero probability mass to every term in the vocabulary for each topic  $t$ . So for computing the intersection, we considered only terms in  $\varphi_t^T$  which have  $p(v|t) = \varphi_{t,v}^T > 0.001$  as probability masses lower than 0.001 are more due to the smoothing effect of the Dirichlet distribution than true correlation. In an actual application, the values for  $k$  and  $q$  can be set according to the user’s need. In our experiment, we used  $k = 3$  and  $q = 5$ , which are reasonable because a post normally does not talk about many topics ( $k$ ), and the contention points (aspect terms) appear quite close to the disagreement expressions.

For comparison, we also designed a baseline. For each disagreement (or question) expression  $w \in d \cap \varphi_{e=Disagreement}^E (\varphi_{e=Question}^E)$ , we emit the nouns and noun phrases within the same window  $q$  as the points of contention (question) in  $d$ . This baseline is reasonable because topical terms are usually nouns and noun phrases and are near disagreement (question) expressions. We note that this baseline cannot stand alone because it has to rely on our expression models  $\Phi^E$  of ME-TME.

Next, to evaluate the performance of these methods in discovering points of contention, we randomly selected 100 disagreement (contentious) (and 100 question) comment posts on reviews from each of the 4 product domains: Ipod, Kindle, Nikon Cameras, and Garmin GPS in our database and employed the aforementioned methods to discover the points of contention (question) in each post. Then we asked two human judges (graduate students fluent in English) to manually judge the results produced by each method for each post. We asked them to report the precision of the discovered terms for a post by judging them as being indeed valid points of contention and report recall in a post by judging how many of actually contentious points in the post were discovered. In Table 5 (a), we report the average precision and recall for 100 posts in each domain by the two judges  $J_1$  and  $J_2$  for different methods on the task of discovering points (aspects) of contention. In Table 5 (b), similar results are reported for the task of discovering questioned aspects in 100 question comments for each product domain. Since this judging task is subjective, the differences in the results from the two judges are not surprising. Our judges were made to work in isolation to prevent any bias. We observe that across all domains, ME-TME again performs the best consistently. Note that agreement study using Kappa is not used here as our problem is not to label a fixed set of items categorically by the judges.

## 6. Conclusion

This paper proposed the problem of modeling review comments, and presented two models TME and ME-TME to model and to extract topics (aspects) and various comment expressions. These expressions enable us to classify comments more accurately, and to find contentious aspects and questioned aspects. These pieces of information also allow us to produce a simple summary of comments for each review as discussed in Section 1. To our knowledge, this is the first attempt to analyze comments in such details. Our experiments demonstrated the efficacy of the models. ME-TME also outperformed TME significantly.

## Acknowledgments

This work is supported in part by National Science Foundation (NSF) under grant no. IIS-1111092.

## References

- Agarwal, R., S. Rajagopalan, R. Srikant, Y. Xu. 2003. Mining newsgroups using networks arising from social behavior. Proceedings of International Conference on World Wide Web 2003.
- Andrzejewski, D., X. Zhu, M. Craven. 2009. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. Proceedings of International Conference on Machine Learning.
- Blei, D., A. Ng, and M. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*.
- Brody, S. and S. Elhadad. 2010. An Unsupervised Aspect-Sentiment Model for Online Reviews. Proceedings of the Annual Conference of the North American Chapter of the ACL.
- Burfoot, C., S. Bird, and T. Baldwin. 2011. Collective Classification of Congressional Floor-Debate Transcripts. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics.
- Galley, M., K. McKeown, J. Hirschberg, E. Shriberg. 2004. Identifying agreement and disagreement in conversational speech: Use of Bayesian networks to model pragmatic dependencies. Proceedings of the 42th Annual Meeting of the Association of Computational Linguistics.
- Ghose, A. and P. Ipeirotis. 2007. Designing novel review ranking systems: predicting the usefulness and impact of reviews. Proceedings of International Conference on Electronic Commerce.
- Griffiths, T. and M. Steyvers. 2004. Finding scientific topics. Proceedings of National Academy of Sciences.
- Kim, S., P. Pantel, T. Chklovski, and M. Pennacchiotti. 2006. Automatically assessing review helpfulness. Proceedings of Empirical Methods in Natural Language Processing.
- Jindal, N. and B. Liu. 2008. Opinion spam and analysis. Proceedings of the ACM International Conference on Web Search and Web Data Mining.
- Jo, Y. and A. Oh. 2011. Aspect and sentiment unification model for online review analysis. Proceedings of the ACM International Conference on Web Search and Web Data Mining.
- Li, F., M. Huang, Y. Yang, and X. Zhu. 2011. Learning to Identify Review Spam. in Proceedings of the International Joint Conference on Artificial Intelligence.
- Lim, E., V. Nguyen, N. Jindal, B. Liu, and H. Lauw. 2010. Detecting Product Review Spammers using Rating Behaviors. Proceedings of the ACM International Conference on Information and Knowledge Management.
- Lin, C. and Y. He. 2009. Joint sentiment/topic model for sentiment analysis. Proceedings of the ACM International Conference on Information and Knowledge Management.
- Liu, J., Y. Cao, C. Lin, Y. Huang, and M. Zhou. 2007. Low-quality product review detection in opinion summarization. Proceedings of Empirical Methods in Natural Language Processing.
- Liu, B. 2012. Sentiment Analysis and Opinion Mining. Morgan & Claypool publishers (to appear in June 2012).
- Liu, Y., X. Huang, A. An, and X. Yu. 2008. Modeling and predicting the helpfulness of online reviews. Proceedings of IEEE International Conference on Data Mining.
- Lu, Y. and C. Zhai. 2008. Opinion integration through semi-supervised topic modeling. Proceedings of International Conference on World Wide Web.
- Lu, Y., C. Zhai, and N. Sundaresan. 2009. Rated aspect summarization of short comments. Proceedings of International Conference on World Wide.
- Mei, Q. X. Ling, M. Wondra, H. Su and C. Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. Proceedings of International Conference on World Wide.
- Moghaddam, S. and M. Ester. 2011. ILDA: interdependent LDA model for learning latent aspects and their ratings from online product reviews. Proceedings of Annual ACM SIGIR Conference on Research and Development in Information Retrieval.
- Mukherjee, A. and B. Liu. 2012a. Aspect Extraction through Semi-Supervised Modeling. Proceedings of 50th Annual Meeting of Association for Computational Linguistics (to appear in July 2012).
- Mukherjee, A. and B. Liu. 2012b. Mining Contentions from Discussions and Debates. Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (to appear in August 2012).
- Mukherjee, A., B. Liu and N. Glance. 2012. Spotting Fake Reviewer Groups in Consumer Reviews. Proceedings of International World Wide Web Conference.
- Murakami A., and R. Raymond, 2010. Support or Oppose? Classifying Positions in Online Debates from Reply Activities and Opinion Expressions. Proceedings of International Conference on

Computational Linguistics.

- O'Mahony, M. P. and B. Smyth. 2009. Learning to recommend helpful hotel reviews. Proceedings of the third ACM conference on Recommender systems.
- Ott, M., Y. Choi, C. Cardie, and J. T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics.
- Pang, B. and L. Lee. 2008. Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval.
- Ramage, D., D. Hall, R. Nallapati, and C. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. Proceedings of Empirical Methods in Natural Language Processing.
- Ramage, D., C. Manning, and S. Dumais. 2011. Partially labeled topic models for interpretable text mining. Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Rosen-Zvi, M., T. Griffiths, M. Steyvers, and P. Smith. 2004. The author-topic model for authors and documents. Uncertainty in Artificial Intelligence.
- Sauper, C. A. Haghighi and R. Barzilay. 2011. Content models with attitude. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics.
- Somasundaran, S., J. Wiebe. 2009. Recognizing stances in online debates. Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP
- Teh, Y., M. Jordan, M. Beal and D. Blei. 2006. Hierarchical Dirichlet Processes. Journal of the American Statistical Association.
- Thomas, M., B. Pang and L. Lee. 2006. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. Proceedings of Empirical Methods in Natural Language Processing.
- Titov, I. and R. McDonald. 2008a. Modeling online reviews with multi-grain topic models. Proceedings of International Conference on World Wide Web.
- Titov, I. and R. McDonald. 2008b. A joint model of text and aspect ratings for sentiment summarization. Proceedings of Annual Meeting of the Association for Computational Linguistics.
- Tsur, O. and A. Rappoport. 2009. Revrank: A fully unsupervised algorithm for selecting the most helpful book reviews. Proceedings of the International AAAI Conference on Weblogs and Social Media.
- Wang, H., Y. Lu, and C. Zhai. 2010. Latent aspect rating analysis on review text data: a rating regression approach. Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.
- Yano, T and N. Smith. 2010. What's Worthy of Comment? Content and Comment Volume in Political Blogs. Proceedings of the International AAAI Conference on Weblogs and Social Media.
- Zhang, Z. and B. Varadarajan. 2006. Utility scoring of product reviews. Proceedings of ACM International Conference on Information and Knowledge Management.
- Zhao, X., J. Jiang, H. Yan, and X. Li. 2010. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. Proceedings of Empirical Methods in Natural Language Processing.

# A Joint Model for Discovery of Aspects in Utterances

**Asli Celikyilmaz**  
Microsoft  
Mountain View, CA, USA  
asli@ieee.org

**Dilek Hakkani-Tur**  
Microsoft  
Mountain View, CA, USA  
dilek@ieee.org

## Abstract

We describe a joint model for understanding user actions in natural language utterances. Our multi-layer generative approach uses both labeled and unlabeled utterances to jointly learn aspects regarding utterance’s target domain (e.g. movies), intention (e.g., finding a movie) along with other semantic units (e.g., movie name). We inject information extracted from unstructured web search query logs as prior information to enhance the generative process of the natural language utterance understanding model. Using utterances from five domains, our approach shows up to 4.5% improvement on domain and dialog act performance over cascaded approach in which each semantic component is learned sequentially and a supervised joint learning model (which requires fully labeled data).

## 1 Introduction

Virtual personal assistance (VPA) is a human to machine dialog system, which is designed to perform tasks such as making reservations at restaurants, checking flight statuses, or planning weekend activities. A typical spoken language understanding (SLU) module of a VPA (Bangalore, 2006; Tur and Mori, 2011) defines a structured representation for utterances, in which the constituents correspond to meaning representations in terms of slot/value pairs (see Table 1). While target *domain* corresponds to the context of an utterance in a dialog, the dialog act represents overall intent of an utterance. The *slots* are entities, which are semantic constituents at the word or phrase level. Learning each component

### Sample utterances on ‘plan a night out’ scenario

(I) Show me theaters in [Austin] playing [iron man 2].  
(II) I’m in the mood for [indian] food tonight, show me the ones [within 5 miles] that have [patios].

### Extracted Class and Labels

Domain	Dialog Act	Slots=Values
(I) Movie	find theater	Location= <i>Austin</i> Movie-Name= <i>iron man 2</i>
(II) Restaurant	find restaurant	Rest-Cuisine= <i>indian</i> Location= <i>within 5 miles</i> Rest-Amenities= <i>patios</i>

Table 1: Examples of utterances with corresponding semantic components, i.e., domain, dialog act, and slots.

is a challenging task not only because there are no *a priori* constraints on what a user might say, but also systems must generalize from a tractably small amount of labeled training data. In this paper, we argue that each of these components are interdependent and should be modeled simultaneously. We build a joint understanding framework and introduce a multi-layer context model for semantic representation of utterances of multiple domains.

Although different strategies can be applied, typically a cascaded approach is used where each semantic component is modeled separately/sequentially (Begeja et al., 2004), focusing less on interrelated aspects, i.e., dialog’s domain, user’s intentions, and semantic tags that can be shared across domains. Recent work on SLU (Jeong and Lee, 2008; Wang, 2010) presents joint modeling of two components, i.e., the domain and slot or dialog act and slot components together. Furthermore, most of these systems rely on labeled training utterances, focusing little on issues such as information sharing between the discourse and word level components across different domains, or variations in use of language. To deal with de-

pendency and language variability issues, a model that considers dependencies between semantic components and utilizes information from large bodies of unlabeled text can be beneficial for SLU.

In this paper, we present a novel generative Bayesian model that learns domain/dialog-act/slot semantic components as latent aspects of text utterances. Our approach can identify these semantic components simultaneously in a hierarchical framework that enables the learning of dependencies. We incorporate prior knowledge that we observe in web search query logs as constraints on these latent aspects. Our model can discover associations between words within a multi-layered aspect model, in which some words are indicative of higher layer (meta) aspects (domain or dialog act components), while others are indicative of lower layer specific entities.

The contributions of this paper are as follows:

- (i) construction of a novel Bayesian framework for semantic parsing of natural language (NL) utterances in a unifying framework in §4,
- (ii) representation of seed labeled data and information from web queries as informative prior to design a novel utterance understanding model in §3 & §4,
- (iii) comparison of our results to supervised sequential and joint learning methods on NL utterances in §5. We conclude that our generative model achieves noticeable improvement compared to discriminative models when labeled data is scarce.

## 2 Background

Language understanding has been well studied in the context of question/answering (Harabagiu and Hickl, 2006; Liang et al., 2011), entailment (Sammons et al., 2010), summarization (Hovy et al., 2005; Daumé-III and Marcu, 2006), spoken language understanding (Tur and Mori, 2011; Dinarelli et al., 2009), query understanding (Popescu et al., 2010; Li, 2010; Reisinger and Pasca, 2011), etc. However data sources in VPA systems pose new challenges, such as variability and ambiguities in natural language, or short utterances that rarely contain contextual information, etc. Thus, SLU plays an important role in allowing any sophisticated spoken dialog system (e.g., DARPA Calo (Berry et al., 2011), Siri, etc.) to take the correct machine actions.

A common approach to building SLU framework

is to model its semantic components separately, assuming that the context (domain) is given *a priori*. Earlier work takes dialog act identification as a classification task to capture the user’s intentions (Margolis et al., 2010) and slot filling as a sequence learning task specific to a given domain class (Wang et al., 2009; Li, 2010). Since these tasks are considered as a pipeline, the errors of each component are transferred to the next, causing robustness issues. Ideally, these components should be modeled simultaneously considering the dependencies between them. For example, in a local domain application, users may require information about a sub-domain (*movies, hotels, etc.*), and for each sub-domain, they may want to take different actions (*find* a movie, *call* a restaurant or *book* a hotel) using domain specific attributes (e.g., *cuisine type* of a restaurant, *titles* for movies or *star-rating* of a hotel). There’s been little attention in the literature on modeling the dependencies of SLU’s correlated structures.

Only recent research has focused on the joint modeling of SLU (Jeong and Lee, 2008; Wang, 2010) taking into account the dependencies at learning time. In (Jeong and Lee, 2008), a triangular chain conditional random fields (Tri-CRF) approach is presented to model two of the SLU’s components in a single-pass. Their discriminative approach represents semantic slots and discourse-level utterance labels (domain or dialog act) in a single structure to encode dependencies. However, their model requires fully labeled utterances for training, which can be time consuming and expensive to generate for dynamic systems. Also, they can only learn dependencies between two components simultaneously.

Our approach differs from the earlier work in that- we take the utterance understanding as a multi-layered learning problem, and build a hierarchical clustering model. Our joint model can discover domain  $D$ , and user’s act  $A$  as higher layer latent concepts of utterances in relation to lower layer latent semantic topics (slots)  $S$  such as named-entities (“*New York*”) or context bearing non-named entities (“*vegan*”). Our work resembles the earlier work of PAM models (Mimno et al., 2007), i.e., directed acyclic graphs representing mixtures of hierarchical topic structures, where upper level topics are multinomial over lower level topics in a hierarchy. In an analogical way to earlier work, the  $D$  and  $A$  in our

approach represent common co-occurrence patterns (dependencies) between semantic tags  $S$  (Fig. 2). Concretely, correlated topics eliminate assignment of semantic tags to segments in an utterance that belong to other domains, e.g., we can discover that "Show me vegan restaurants in San Francisco" has a low probability of outputting a *movie-actor* slot. Being generative, our model can incorporate unlabeled utterances and encode prior information of concepts.

### 3 Data and Approach Overview

Here we define several abstractions of our joint model as depicted in Fig. 1. Our corpus mainly contains NL utterances ("show me the nearest dinner places") and some keyword queries ("iron man 2 trailers"). We represent each utterance  $u$  as a vector  $w_u$  of  $N_u$  word n-grams (segments),  $w_{uj}$ , each of which are chosen from a vocabulary  $W$  of fixed-size  $V$ . We use entity lists obtained from web sources (explained next) to identify segments in the corpus. Our corpus contains utterances from  $K_D=4$  main **domains**:  $\in \{movies, hotels, restaurants, events\}$ , as well as out-of-domain *other* class. Each utterance has one **dialog act** ( $A$ ) associated with it. We assume a fixed number of possible dialog acts  $K_A$  for each domain. Semantic Tags, **slots** ( $S$ ) are lexical units (segments) of an utterance, which we classify into two types: domain-independent slots that are shared across all domains, (e.g., *location, time, year, etc.*), and domain-dependent slots, (e.g. *movie-name, actor-name, restaurant-name, etc.*). For tractability, we consider a fixed number of latent slot types  $K_S$ . Our algorithm assigns domain/dialog-act/slot labels to each topic at each layer in the hierarchy using labeled data (explained in §4.)

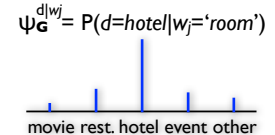
We represent domain and dialog act components as meta-variables of utterances. This is similar to author-topic models (Rosen-Zvi et al., 2004), that capture author-topic relations across documents. In that case, words are generated by first selecting an author uniformly from an observed author list and then selecting a topic from a distribution over words that is specific to that author. In our model, each utterance  $u$  is associated with domain and dialog act topics. A word  $w_{uj}$  in  $u$  is generated by first selecting a domain and an act topic and then slot topic over words of  $u$ . The domain-dependent slots

in utterances are usually not dependent on the dialog act. For instance, while "find [hugo] trailer" and "show me where [hugo] is playing" have both a movie-name slot ("hugo"), they have different dialog acts, i.e., find-trailer and find-movie, respectively. We predict posterior probabilities for domain  $\tilde{P}(d \in D|u)$  dialog act  $\tilde{P}(a \in A|ud)$  and slots  $\tilde{P}(s_j \in S|w_{uj}, d, s_{j-1})$  of words  $w_{uj}$  in sequence.

To handle language variability, and hence discover correlation between hierarchical aspects of utterances<sup>1</sup>, we extract prior information from two web resources as follows:

**Web n-Grams (G).** Large-scale engines such as Bing or Google log more than 100M search queries each day. Each query in the search logs has an associated set of URLs that were clicked after users entered a given query. The click information can be used to infer domain class labels, and therefore, can provide (noisy) supervision in training domain classifiers. For example, two queries ("cheap hotels Las Vegas" and "wine resorts in Napa"), which resulted in clicks on the same base URL (e.g., www.hotels.com) probably belong to the same domain ("hotels" in this case).

Given query logs, we compile sets of in-domain queries based on their base URLs<sup>2</sup>. Then, for each vocabulary item  $w_j \in W$  in our corpus, we calculate frequency of  $w_j$  in each set of in-domain queries and represent each word (e.g., "room") as a discrete normalized probability distribution  $\psi_G^j$  over  $K_D$  domains  $\{\psi_G^{dj}\} \in \psi_G^j$ . We inject them as nonuniform priors over domain and dialog act parameters in §4.



**Entity Lists (E).** We limit our model to a set of *named-entity* slots (e.g., *movie-name, restaurant-name*) and *non-named entity* slots (e.g., *restaurant-cuisine, hotel-rating*). For each entity slot, we extract a large collection of entity lists through the url's on the web that correspond to our domains, such as movie-names listed on IMDB, restaurant-names on OpenTable, or hotel-ratings on tripadvisor.com.

<sup>1</sup>Two utterances can be intrinsically related but contain no common terms, e.g., "has open bar" and "serves free drinks".

<sup>2</sup>We focus on domain specific search engines such as IMDB.com, RottenTomatoes.com for movies, Hotels.com and Expedia.com for hotels, etc.



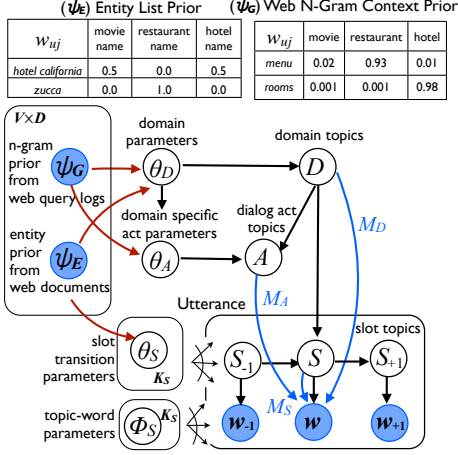


Figure 1: Graphical model depiction of the MCM.  $\mathbf{D}$ ,  $\mathbf{A}$ ,  $\mathbf{S}$  are domain, dialog act and slot in a hierarchy, each consisting of  $K_D, K_A, K_S$  components. Shaded nodes indicate observed variables. Hyper-parameters are omitted. Sample informative priors over latent topics  $\psi_E$  and  $\psi_G$  are shown. Blue arrows indicate frequency of vocabulary terms sampled for each topic.

We represent each entity list as observed nonuniform priors  $\psi_E$  and inject them into our joint learning process as  $V$  sparse multinomial distributions over latent topics  $D$ , and  $S$  to "guide" the generation of utterances (Fig. 1 top-left table), explained in §4.

#### 4 Multi-Layer Context Model - MCM

The generative process of our multi-layer context model (MCM) (Fig. 1) is shown in Algorithm 1. Each utterance  $u$  is associated with  $d = 1..K_D$  multinomial domain-topic distributions  $\theta_D^d$ . Each domain  $d$ , is represented as a distribution over  $a = 1, \dots, K_A$  dialog acts  $\theta_A^{da}$  ( $\theta_D^d \rightarrow \theta_A^{da}$ ). In our MCM model, we assume that each utterance is represented as a hidden Markov model with  $K_S$  slot states. Each state generates n-grams according to a multinomial n-gram distribution. Once domain  $D_u$  and act  $A_{ud}$  topics are sampled for  $u$ , a slot state topic  $S_{ujd}$  is drawn to generate each segment  $w_{uj}$  of  $u$  by considering the word-tag sequence frequencies based on a simple HMM assumption, similar to the content models of (Sauper et al., 2011). Initial and transition probability distributions over the HMM states are sampled from Dirichlet distribution over slots  $\theta_S^{ds}$ . Each slot state  $s$  generates words according to multinomial word distribution  $\phi_S^s$ . We also keep track of the frequency of vocabulary terms  $w_j$ 's in a  $V \times K_D$  matrix  $M_D$ . Every time a  $w_j$  is sampled for a domain  $d$ , we increment its count, a degree of domain bearing

words. Similarly, we keep track of dialog act and slot bearing words in  $V \times K_A$  and  $V \times K_S$  matrices,  $M_A$  and  $M_S$  (shown as red arrows in Fig 1). Being Bayesian, each distribution  $\theta_D^d$ ,  $\theta_A^{da}$ , and  $\theta_S^{ds}$  is sampled from a Dirichlet prior distribution with different parameters, described next.

#### Algorithm 1 Multi-Layer Context Model Generation

- 1: **for** each domain  $d \leftarrow 1, \dots, K_D$
- 2:   draw domain dist.  $\theta_D^d \sim \text{Dir}(\alpha_D^*)^\dagger$ ,
- 3:   **for** each dialog-act  $a \leftarrow 1, \dots, K_A$
- 4:     draw dialog act dist.  $\theta_A^{da} \sim \text{Dir}(\alpha_A^*)$ ,
- 5:     **for** each slot type  $s \leftarrow 1, \dots, K_S$
- 6:       draw slot dist.  $\theta_S^{ds} \sim \text{Dir}(\alpha_S^*)$ .
- 7:   **endfor**
- 8: draw  $\phi_S^s \sim \text{Dir}(\beta)$  for each slot type  $s \leftarrow 1, \dots, K_S$ .
- 9: **for** each utterance  $u \leftarrow 1, \dots, |U|$  **do**
- 10:   Sample a domain  $D_u \sim \text{Multi}(\theta_D^d)$  and,
- 11:   and act topic  $A_{ud} \sim \text{Multi}(\theta_A^{da})$ .
- 12:   **for** words  $w_{uj}, j \leftarrow 1, \dots, N_u$  **do**
- 13:     - Draw  $S_{ujd} \sim \text{Multi}(\theta_S^{D_u, S_{u(j-1)d}})^\ddagger$ .
- 14:     - Sample  $w_{uj} \sim \text{Multi}(\phi_S^{S_{ujd}})$ .
- 15:   **endfor**
- 16: **endfor**

$\dagger \text{Dir}(\alpha_D^*), \text{Dir}(\alpha_A^*), \text{Dir}(\alpha_S^*)$  are parameterized based on prior knowledge.

$\ddagger$  Here HMM assumption over utterance words is used.

In hierarchical topic models (Blei et al., 2003; Mimno et al., 2007), etc., topics are represented as distributions over words, and each document expresses an admixture of these topics, both of which have symmetric Dirichlet ( $\text{Dir}$ ) prior distributions. Symmetric Dirichlet distributions are often used, since there is typically no prior knowledge favoring one component over another. In the topic model literature, such constraints are sometimes used to deterministically allocate topic assignments to known labels (Labeled Topic Modeling (Ramage et al., 2009)) or in terms of pre-learned topics encoded as prior knowledge on topic distributions in documents (Reisinger and Paşca, 2009). Similar to previous work, we define a latent topic per each known semantic component label, e.g., five domain topics for five defined domains. Different from earlier work though, we also inject knowledge that we extract from several resources including entity lists from web search query click logs as well as seed labeled training utterances as prior information. We constrain the generation of the semantic components of our model by encoding prior knowledge in terms of

asymmetric Dirichlet topic priors  $\alpha=(\alpha m_1, \dots, \alpha m_K)$  where each  $k$ th topic has a prior weight  $\alpha_k=\alpha m_k$ , with varying base measure  $\mathbf{m}=(m_1, \dots, m_k)$ <sup>3</sup>.

We update parameter vectors of Dirichlet domain prior  $\alpha_D^{u*}=\{(\alpha_D \cdot \psi_D^{u1}), \dots, (\alpha_D \cdot \psi_D^{uK_D})\}$ , where  $\alpha_D$  is the concentration parameter for domain Dirichlet distribution and  $\psi_D^u=\{\psi_D^{ud}\}_{d=1}^{K_D}$  is the base measure which we obtain from various resources. Because base measure updates are dependent on prior knowledge of corpus words, each utterance  $u$  gets a different base measure. Similarly, we update the parameter vector of the Dirichlet dialog act and slot priors  $\alpha_A^{u*}=\{(\alpha_A \cdot \psi_A^{u1}), \dots, (\alpha_A \cdot \psi_A^{uK_A})\}$  and  $\alpha_S^{u*}=\{(\alpha_S \cdot \psi_S^{u1}), \dots, (\alpha_S \cdot \psi_S^{uK_S})\}$  using base measures  $\psi_A^u=\{\psi_A^{ua}\}_{a=1}^{K_A}$  and  $\psi_S^u=\{\psi_S^{us}\}_{s=1}^{K_S}$  respectively.

Before describing base measure update for domain, act and slot Dirichlet priors, we explain the constraining prior knowledge parameters below:

★ **Entity List Base Measure** ( $\psi_E^j$ ): Entity features are indicative of domain and slots and MCM utilizes these features while sampling topics. For instance, entities *hotel-name* "Hilton" and *location* "New York" are discriminative features in classifying "find nice cheap double room in New York Hilton" into correct domain (*hotel*) and slot (*hotel-name*) clusters. We represent entity lists corresponding to known domains as multinomial distributions  $\psi_E^j$ , where each  $\psi_E^{dj}$  is the probability of entity-word  $w_j$  used in the domain  $d$ . Some entities may belong to more than one domain, e.g., "hotel California" can either be a movie, or song or hotel name.

★ **Web n-Gram Context Base Measure** ( $\psi_G^j$ ): As explained in §3, we use the web n-grams as additional information for calculating the base measures of the Dirichlet topic distributions. Normalized word distributions  $\psi_G^j$  over domains were used as weights for domain and dialog act base measure.

★ **Corpus n-Gram Base Measure** ( $\psi_C^j$ ): Similar to other measures, MCM also encodes n-gram constraints as word-frequency features extracted from labeled utterances. Concretely, we calculate the frequency of vocabulary items given domain-act label pairs from the training labeled utterances and convert there into probability measures over domain-acts. We encode conditional

probabilities  $\{\psi_C^{adj}\} \in \psi_C^j$  as multinomial distributions of words over domain-act pairs, e.g.,  $\psi_C^{adj} = P(d="restaurant", a="make-reservation"|"table")$ .

**Base measure update:** The  $\alpha$ -base measures are used to shape Dirichlet priors  $\alpha_D^{u*}$ ,  $\alpha_A^{u*}$  and  $\alpha_S^{u*}$ . We update the base measures of each sampled domain  $D_u = d$  given each vocabulary  $w_j$  as:

$$\psi_D^{dj} = \begin{cases} \psi_E^{dj}, & \psi_E^{dj} > 0 \\ \psi_G^{dj}, & \text{otherwise} \end{cases} \quad (1)$$

In (1) we assume that entities ( $E$ ) are more indicative of the domain compared to other n-grams ( $G$ ) and should be more dominant in sampling decision for domain topics. Given an utterance  $u$ , we calculate its base measure  $\psi_D^{ud} = (\sum_j^{N_u} \psi_D^{dj}) / N_u$ .

Once the domain is sampled, we update the prior weight of dialog acts  $A_{ud} = a$ :

$$\psi_A^{aj} = \psi_C^{adj} * \psi_G^{dj} \quad (2)$$

and slot components  $S_{ujd} = s$ :

$$\psi_S^{sj} = \psi_E^{dj} \quad (3)$$

Then we update their base measures for a given  $u$  as:  $\psi_A^{ua} = (\sum_j^{N_u} \psi_A^{aj}) / N_u$  and  $\psi_S^{us} = (\sum_j^{N_u} \psi_S^{sj}) / N_u$ .

#### 4.1 Inference and Learning

The goal of inference is to predict the domain, user's act and slot distributions over each segment given an utterance. The MCM has the following set of parameters: domain-topic distributions  $\theta_D^d$  for each  $u$ , the act-topic distributions  $\theta_A^{da}$  for each domain topic  $d$  of  $u$ , local slot-topic distributions for each domain  $\theta^S$ , and  $\phi_S^s$  for slot-word distributions. Previous work (Asuncion et al., 2009; Wallach et al., 2009) shows that the choice of inference method has negligible effect on the probability of testing documents or inferred topics. Thus, we use Markov Chain Monte Carlo (MCMC) method, specifically Gibbs sampling, to model the posterior distribution  $P_{\text{MCM}}(D_u, A_{ud}, S_{ujd} | \alpha_D^{u*}, \alpha_A^{u*}, \alpha_S^{u*}, \beta)$  by obtaining samples  $(D_u, A_{ud}, S_{ujd})$  drawn from this distribution. For each utterance  $u$ , we sample a domain  $D_u$  and act  $A_{ud}$  and hyper-parameters  $\alpha_D$  and  $\alpha_A$  and their base measures  $\psi_D^{ud}$ ,  $\psi_A^{ua}$  (from Eq. 1,2):

$$\theta_D^d = \frac{N_u^d + \alpha_D \psi_D^{ud}}{N_u + \alpha_D^{u*}}; \quad \theta_A^{da} = \frac{N_{a|ud} + \alpha_A \psi_A^{ua}}{N_{ud} + \alpha_A^{u*}} \quad (4)$$

The  $N_u^d$  is the number of occurrences of domain topic  $d$  in utterance  $u$ ,  $N_{a|ud}$  is the number of occurrences of act  $a$  given  $d$  in  $u$ . During sampling of a

<sup>3</sup>See (Wallach, 2008) Chapter 3 for analysis of hyper-priors on topic models.

slot state  $S_{ujd}$ , we assume that utterance is generated by the HMM model associated with the assigned domain. For each segment  $w_{uj}$  in  $u$ , we sample a slot state  $S_{ujd}$  given the remaining slots and hyperparameters  $\alpha_S$ ,  $\beta$  and base measure  $\psi_S^{us}$  (Eq. 3) by:

$$p(S_{ujd} = s | \mathbf{w}, \mathbf{D}_u, \mathbf{S}_{-(ujd)} \alpha_S^{\mathbf{u}^*}, \beta) \propto \frac{N_{ujd}^k + \beta}{N_{(\cdot)}^k + V\beta} * (N_s^{D_u, S_{u(j-1)d}} + \alpha_S \psi_S^{us}) * \frac{N_{S_{u(j+1)d}}^{D_u, s} + \mathbb{I}(S_{uj-1}, s) + \mathbb{I}(S_{uj+1}, s) + \alpha_S \psi_S^{us}}{N_{(\cdot)}^{D_u, s} + \mathbb{I}(S_{uj-1}, s) + K_D \alpha_S^{\mathbf{u}^*}} \quad (5)$$

The  $N_{ujd}^k$  is the number of times segment  $w_{uj}$  is generated from slot state  $s$  in all utterances assigned to domain topic  $d$ ,  $N_{s_2}^{D_u, s_1}$  is the number of transitions from slot state  $s_1$  to  $s_2$ , where  $s_1 \in \{S_{u(j-1)d}, S_{u(j+1)d}\}$ ,  $\mathbb{I}(s_1, s_2) = 1$  if slot  $s_1 = s_2$ .

## 4.2 Semantic Structure Extraction with MCM

During Gibbs sampling, we keep track of the frequency of draws of domain, dialog act and slot indicating n-grams  $w_j$ , in  $M_D$ ,  $M_A$  and  $M_S$  matrices, respectively. These n-grams are context bearing words (examples are shown in Fig.1.). For given  $u$  the predicted domain  $d_u^*$  is determined by:

$$d_u^* = \arg \max_d \tilde{P}(d|u) = \arg \max_d [\theta_D^d * \prod_{j=1}^{N_u} \frac{M_D^{jd}}{M_D}]$$

and predicted dialog act by  $\arg \max_a \tilde{P}(a|ud^*)$ :

$$a_u^* = \arg \max_a [\theta_A^{d^* a} * \prod_{j=1}^{N_u} \frac{M_A^{ja}}{M_A}] \quad (6)$$

For each segment  $w_{uj}$  in  $u$ , its predicted slot are determined by  $\arg \max_s P(s_j | w_{uj}, d^*, s_{j-1})$ :

$$s_{uj}^* = \arg \max_s [p(S_{ujd^*} = s | \cdot) * \prod_{j=1}^{N_u} \frac{Z_S^{js}}{Z_S}] \quad (7)$$

## 5 Experiments

We performed several experiments to evaluate our proposed approach. Before presenting our results, we describe our datasets as well as two baselines.

### 5.1 Datasets, Labels and Tags

Our dataset contains utterances obtained from dialogs between human users and our personal assistant system. We use the transcribed text forms of

Domain	Sample Dialog Acts (DAs) & Slots
movie	<b>DAs:</b> find-movie/director/actor, buy-ticket <b>Slots:</b> name, mpaa-rating ( <i>g-rated</i> ), date, director/actor-name, award( <i>oscar winning</i> )...
hotel	<b>DAs:</b> find-hotel, book-hotel, <b>Slots:</b> name, room-type( <i>double</i> ), amenities, smoking, reward-program( <i>platinum elite</i> )...
restaurant	<b>DAs:</b> find-restaurant, make-reservation, <b>Slots:</b> opening-hour, amenities, meal-type,...
event	<b>DAs:</b> find-event/ticket/performers, get-info.. <b>Slots:</b> name, type( <i>concert</i> ), performer...

Table 2: List of domains, dialog acts and semantic slot tags of utterance segments. Examples for some slots values are presented in parenthesis as *italicized*.

the utterances obtained from (acoustic modeling engine) to train our models<sup>4</sup>. Thus, our dataset contains 18084 NL utterances, 5034 of which are used for measuring the performance of our models. The dataset consists of five domain classes, i.e. *movie*, *restaurant*, *hotel*, *event*, *other*, 42 unique dialog acts and 41 slot tags. Each utterance is labeled with a domain, dialog act and a sequence of slot tags corresponding to segments in utterance (see examples in Table 1). Table 2 shows sample dialog act and slot labels. Annotation agreement, Kappa measure (Cohen, 1960), was around 85%.

We pulled a month of web query logs and extracted over 2 million search queries from the movie, hotel, event, and restaurant domains. We also used generic web queries to compile a set of 'other' domain queries. Our vocabulary consists of n-grams and segments (phrases) in utterances that are extracted using web n-grams and entity lists of §3. We extract distributions of n-grams and entities to inject as prior weights for entity list base ( $\psi_{\mathbf{E}}^j$ ) and web n-gram context base measures ( $\psi_{\mathbf{G}}^j$ ) (see §4).

### 5.2 Baselines and Experiment Setup

We evaluated two baselines and two variants of our joint SLU approach as follows:

★ **Sequence-SLU:** A traditional approach to SLU extracts domain, dialog act and slots as semantic components of utterances using three sequential models. Typically, domain and dialog act detection models are taken as query classification, where a given NL query is assigned domain and act labels. Among supervised query classification meth-

<sup>4</sup>We submitted sample utterances used in our models as additional resource. Due to licensing issues, we will reveal the full train/test utterances upon acceptance of our paper.

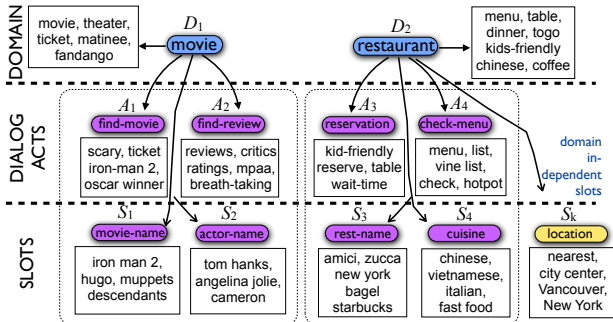


Figure 2: Sample topics discovered by Multi-Layer Context Model (MCM). Given samples of utterances, MCM is able to infer a meaningful set of dialog act ( $A$ ) and slots ( $S$ ), falling into broad categories of domain classes ( $D$ ).

ods, we used the `Adaboost`, utterance classification method that starts from a set of weak classifiers and builds a strong classifier by boosting the weak classifiers. Slot discovery is taken as a sequence labeling task in which *segments* in utterances are labeled (Li, 2010). For segment labeling we use Semi-Markov Conditional Random Fields (`Semi-CRF`) (Sarawagi and Cohen, 2004) method as a benchmark in evaluating semantic tagging performance.

★ **Tri-CRF**: We used Triangular Chain CRF (Jeong and Lee, 2008) as our supervised joint model baseline. It is a state-of-the-art method that learns the sequence labels and utterance class (domain or dialog act) as meta-sequence in a joint framework. It encodes the inter-dependence between the slot sequence  $s$  and meta-sequence label ( $d$  or  $a$ ) using a triangular chain (dual-layer) structure.

★ **Base-MCM**: Our first version injects an informative prior for domain, dialog act and slot topic distributions using information extracted from only labeled training utterances and inject as prior constraints (corpus  $n$ -gram base measure  $\psi_C^j$ ) during topic assignments.

★ **WebPrior-MCM**: Our full model encodes distributions extracted from labeled training data as well as structured web logs as asymmetric Dirichlet priors. We analyze performance gain by the information from web sources ( $\psi_G^j$  and  $\psi_E^j$ ) when injected into our approach compared to `Base-MCM`.

We inject dictionary constraints as features to train supervised discriminative methods, i.e., boosting and `Semi-CRF` in `Sequence-SLU`, and `Tri-CRF` models. For semantic tagging, dictionary constraints apply to the features between individual

segments and their labels, and for utterance classification (to predict domain and dialog acts) they apply to the features between utterance and its label. Given a list of dictionaries, these constraints specify which label is more likely. For discriminative methods, we use several named entities, e.g., `Movie-Name`, `Restaurant-Name`, `Hotel-Name`, etc., non-named entities, e.g., `Genre`, `Cuisine`, etc., and domain independent dictionaries, e.g., `Time`, `Location`, etc.

We train domain and dialog act classifiers via `Icsiboost` (Favre et al., 2007) with 10K iterations using lexical features (up to 3-n-grams) and constraining dictionary features (all dictionaries). For feature templates of sequence learners, i.e., `Semi-CRF` and `Tri-CRF`, we use current word, bi-gram and dictionary features. For `Base-MCM` and `WebPrior-MCM`, we run Gibbs sampler for 2000 iterations with the first 500 samples as burn-in.

### 5.3 Evaluations and Discussions

We evaluate the performance of our joint model on two experiments using two metrics. For domain and dialog act detection performance we present results in accuracy, and for slot detection we use the F1 pairwise measure.

**Experiment 1. Encoding Prior Knowledge:** A common evaluation method in SLU tasks is to measure the performance of each individual semantic model, i.e., domain, dialog act and semantic tagging (slot filling). Here, we not only want to demonstrate the performance of each component of MCM but also their performance under limited amount of labeled data. We randomly select subsets of labeled training data  $U_L^i \subset U_L$  with different samples sizes,  $n_L^i = \{\gamma * n_L\}$ , where  $n_L$  represents the sample size of  $U_L$  and  $\gamma = \{10\%, 25\%, \dots\}$  is the subset percentage. At each random selection, the rest of the utterances are used as unlabeled data to boost the performance of MCM. The supervised baselines do not leverage the unlabeled utterances.

The results reported in Figure 3 reveal both the strengths and some shortcomings of our approach. When the number of labeled data is small ( $n_L^i \leq 25\% * n_L$ ), our `WebPrior-MCM` has a better performance on domain and act predictions compared to the two baselines. Compared to `Sequence-SLU`, we observe 4.5% and 3% performance improvement on the domain and dialog act

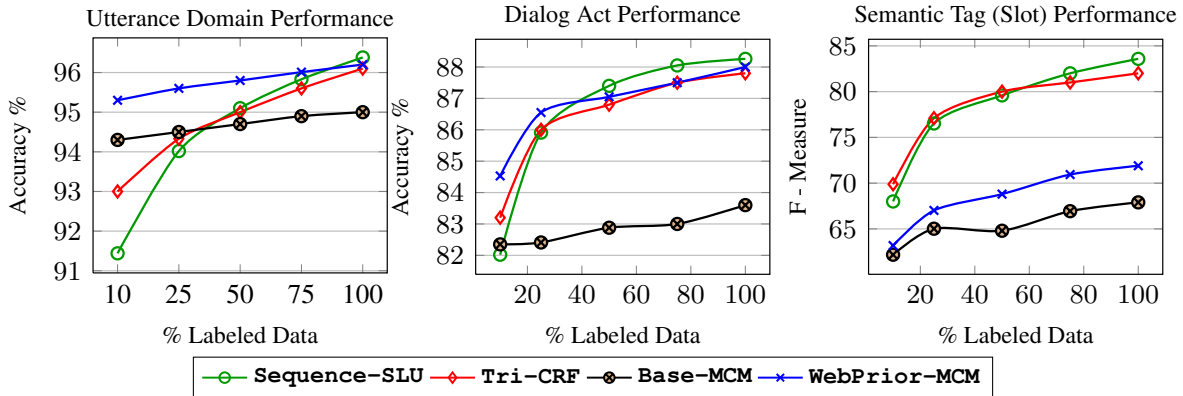


Figure 3: Semantic component extraction performance measures for various baselines as well as our approach with different priors.

models, whereas our gain is 2.6% and 1.7% over Tri-CRF models. As the percentage of labeled utterances in training data increase, Tri-CRF performance increases, however WebPrior-MCM is still comparable with Sequence-SLU. This is because we utilize domain priors obtained from the web sources as supervision during generative process as well as unlabeled utterances that enable handling language variability. Adding labeled data improves the performance of all models however supervised models benefit more compared to MCM models.

Although WebPrior-MCM’s domain and dialog act performances are comparable (if not better than) the other baselines, it falls short on the semantic tagging model. This is partially due to the HMM assumption compared to the supervised conditional model’s used in the other baselines, i.e., Semi-CRF in Sequence-SLU and Tri-CRF). Our work can be extended by replacing HMM assumption with CRF based sequence learner to enhance the capability of the sequence tagging component of MCM.

**Experiment 2. Less is More?** Being Bayesian, our model can incorporate unlabeled data at training time. Here, we evaluate the performance gain on domain, act and slot predictions as more unlabeled data is introduced at learning time. We use only 10% of the utterances as labeled data in this experiment and incrementally add unlabeled data (90% of labeled data are treated as unlabeled).

The results are shown in Table 3.  $n\%$  ( $n=10,25,\dots$ ) unlabeled data indicates that the WebPrior-MCM is trained using  $n\%$  of unlabeled utterances along with training utterances. Adding unlabeled data has a positive impact on the performance of all three se-

Table 3: Performance evaluation results of WebPrior-MCM using different sizes of unlabeled utterances at learning time.

Unlabeled %	Domain Accuracy	Dialog Act Accuracy	Slot F-Measure
10%	94.69	84.17	52.61
25%	94.89	84.29	54.22
50%	95.08	84.39	56.58
75%	95.19	84.44	<b>57.45</b>
100%	<b>95.28</b>	84.52	<b>58.18</b>

semantic components when WebPrior-MCM is used. The results show that our joint modeling approach has an advantage over the other joint models (i.e., Tri-CRF) in that it can leverage unlabeled NL utterances. Our approach might be usefully extended into the area of understanding search queries, where an abundance of unlabeled queries is observed.

## 6 Conclusions

In this work, we introduced a joint approach to spoken language understanding that integrates two properties (*i*) identifying user actions in multiple domains in relation to semantic units, (*ii*) utilizing large amounts of unlabeled web search queries that suggest the user’s hidden intentions. We proposed a semi-supervised generative joint learning approach tailored for injecting prior knowledge to enhance the semantic component extraction from utterances as a unifying framework. Experimental results using the new Bayesian model indicate that we can effectively learn and discover meta-aspects in natural language utterances, outperforming the supervised baselines, especially when there are fewer labeled and more unlabeled utterances.



## References

- A. Asuncion, M. Welling, P. Smyth, and Y. W. Teh. 2009. On smoothing and inference for topic models. *UAI*.
- S. Bangalore. 2006. Introduction to special issue of spoken language understanding in conversational systems. In *Speech Conversation*, volume 48, pages 233–238.
- L. Begeja, B. Renger, Z. Liu D. Gibbon, and B. Shahraray. 2004. Interactive machine learning techniques for improving slu models. In *Proceedings of the HLT-NAACL 2004 Workshop on Spoken Language Understanding for Conversational Systems and Higher Level Linguistic Information for Speech Processing*.
- Pauline M. Berry, Melinda Gervasio, Bart Peintner, and Neil Yorke-Smith. 2011. Ptime: Personalized assistance for calendaring. In *ACM Transactions on Intelligent Systems and Technology*, volume 2, pages 1–40.
- D. Blei, A. Ng, and M. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.
- J. Cohen. 1960. A coefficient of agreement for nominal scales. In *Educational and Psychological Measurement*, volume 20, pages 37–46.
- H. Daumé-III and D. Marcu. 2006. Bayesian query focused summarization.
- M. Dinarelli, A. Moschitti, and G. Riccardi. 2009. Re-ranking models for spoken language understanding. *Proc. European Chapter of the Annual Meeting of the Association of Computational Linguistics (EACL)*.
- B. Favre, D. Hakkani-Tür, and Sebastien Cuendet. 2007. Icsiboost. <http://code.google.com/p/icsiboost>.
- S. Harabagiu and A. Hickl. 2006. Methods for using textual entailment for question answering. pages 905–912.
- E. Hovy, C.Y. Lin, and L. Zhou. 2005. A be-based multi-document summarizer with query interpretation. *Proc. DUC*.
- M. Jeong and G. G. Lee. 2008. Triangular-chain conditional random fields. *EEE Transactions on Audio, Speech and Language Processing (IEEE-TASLP)*.
- X. Li. 2010. Understanding semantic structure of noun phrase queries. *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*.
- P. Liang, M. I. Jordan, and D. Klein. 2011. Learning dependency based compositional semantics.
- A. Margolis, K. Livescu, and M. Osterdorf. 2010. Domain adaptation with unlabeled data for dialog act tagging. In *Proc. Workshop on Domain Adaptation for Natural Language Processing at the the Annual Meeting of the Association of Computational Linguistics (ACL)*.
- D. Mimno, W. Li, and A. McCallum. 2007. Mixtures of hierarchical topics with pachinko allocation. *Proc. ICML*.
- A. Popescu, P. Pantel, and G. Mishne. 2010. Semantic lexicon adaptation for use in query interpretation. *19th World Wide Web Conference (WWW-10)*.
- D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. 2009. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. *Proc. EMNLP*.
- J. Reisinger and M. Paşca. 2009. Latent variable models of concept-attribute attachment. *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*.
- J. Reisinger and M. Pasca. 2011. Fine-grained class label markup of search queries. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*.
- M. Sammons, V. Vydiswaran, and D. Roth. 2010. Ask not what textual entailment can do for you... In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*, Uppsala, Sweden, 7.
- S. Sarawagi and W. W. Cohen. 2004. Semimarkov conditional random fields for information extraction. *Proc. NIPS*.
- C. Sauper, A. Haghighi, and R. Barzilay. 2011. Content models with attitude. In *Proc. of the Annual Meeting of the Association of Computational Linguistics (ACL)*.
- G. Tur and R. De Mori. 2011. Spoken language understanding: Systems for extracting semantic information from speech. *Wiley*.
- H. Wallach, D. Mimno, and A. McCallum. 2009. Rethinking lda: Why priors matter. *NIPS*.
- H. Wallach. 2008. Structured topic models for language. *Ph.D. Thesis, University of Cambridge*.
- Y.Y. Wang, R. Hoffman, X. Li, and J. Szymanski. 2009. Semi-supervised learning of semantic classes for query understanding from the web and for the web. In *The 18th ACM Conference on Information and Knowledge Management*.
- Y-Y. Wang. 2010. Strategies for statistical spoken language understanding with small amount of data - an empirical study. *Proc. Interspeech 2010*.

# Aspect Extraction through Semi-Supervised Modeling

**Arjun Mukherjee**

Department of Computer Science  
University of Illinois at Chicago  
Chicago, IL 60607, USA  
arjun4787@gmail.com

**Bing Liu**

Department of Computer Science  
University of Illinois at Chicago  
Chicago, IL 60607, USA  
liub@cs.uic.edu

## Abstract

Aspect extraction is a central problem in sentiment analysis. Current methods either extract aspects without categorizing them, or extract and categorize them using unsupervised topic modeling. By categorizing, we mean the synonymous aspects should be clustered into the same category. In this paper, we solve the problem in a different setting where the user provides some seed words for a few aspect categories and the model extracts and clusters aspect terms into categories simultaneously. This setting is important because categorizing aspects is a subjective task. For different application purposes, different categorizations may be needed. Some form of user guidance is desired. In this paper, we propose two statistical models to solve this seeded problem, which aim to discover exactly what the user wants. Our experimental results show that the two proposed models are indeed able to perform the task effectively.

## 1 Introduction

Aspect-based sentiment analysis is one of the main frameworks for sentiment analysis (Hu and Liu, 2004; Pang and Lee, 2008; Liu, 2012). A key task of the framework is to extract aspects of entities that have been commented in opinion documents. The task consists of two sub-tasks. The first sub-task extracts *aspect terms* from an opinion corpus. The second sub-task clusters synonymous aspect terms into categories where each category

represents a single aspect, which we call an *aspect category*. Existing research has proposed many methods for aspect extraction. They largely fall into two main types. The first type only extracts aspect terms without grouping them into categories (although a subsequent step may be used for the grouping, see Section 2). The second type uses statistical topic models to extract aspects and group them at the same time in an unsupervised manner. Both approaches are useful. However, in practice, one also encounters another setting, where grouping is not straightforward because for different applications the user may need different groupings to reflect the application needs. This problem was reported in (Zhai et al., 2010), which gave the following example. In car reviews, internal design and external design can be regarded as two separate aspects, but can also be regarded as one aspect, called “design”, based on the level of details that the user wants to study. It is also possible that the same word may be put in different categories based on different needs. However, (Zhai et al., 2010) did not extract aspect terms. It only categorizes a set of given aspect terms.

In this work, we propose two novel statistical models to extract and categorize aspect terms automatically given some seeds in the user interested categories. It is thus able to best meet the user’s specific needs. Our models also jointly model both aspects and aspect specific sentiments. The first model is called SAS and the second model is called ME-SAS. ME-SAS improves SAS by using Maximum-Entropy (or Max-Ent for short) priors to help separate aspects and sentiment terms. However, to train Max-Ent, we do not need manually labeled training data (see Section 4).

In practical applications, asking users to provide some seeds is easy as they are normally experts in their trades and have a good knowledge what are important in their domains.

Our models are related to topic models in general (Blei et al., 2003) and joint models of aspects and sentiments in sentiment analysis in specific (e.g., Zhao et al., 2010). However, these current models are typically unsupervised. None of them can use seeds. With seeds, our models are thus semi-supervised and need a different formulation. Our models are also related to the DF-LDA model in (Andrzejewski et al., 2009), which allows the user to set must-link and cannot-link constraints. A must-link means that two terms must be in the same topic (aspect category), and a cannot-link means that two terms cannot be in the same topic. Seeds may be expressed with must-links and cannot-links constraints. However, our models are very different from DF-LDA. First of all, we jointly model aspect and sentiment, while DF-LDA is only for topics/aspects. Joint modeling ensures clear separation of aspects from sentiments producing better results. Second, our way of treating seeds is also different from DF-LDA. We discuss these and other related work in Section 2.

The proposed models are evaluated using a large number of hotel reviews. They are also compared with two state-of-the-art baselines. Experimental results show that the proposed models outperform the two baselines by large margins.

## 2 Related Work

There are many existing works on aspect extraction. One approach is to find frequent noun terms and possibly with the help of dependency relations (Hu and Liu, 2004; Popescu and Etzioni, 2005; Zhuang et al., 2006; Blair-Goldensohn et al., 2008; Ku et al., 2006; Wu et al., 2009; Somasundaran and Wiebe, 2009; Qiu et al., 2011). Another approach is to use supervised sequence labeling (Liu, Hu and Cheng 2005; Jin and Ho, 2009; Jakob and Gurevych, 2010; Li et al., 2010; Choi and Cardie, 2010; Kobayashi et al., 2007; Yu et al., 2011). Ma and Wan (2010) also exploited centering theory, and (Yi et al., 2003) used language models. However, all these methods do not group extracted aspect terms into categories. Although there are works on grouping aspect terms (Carenini et al., 2005; Zhai et al., 2010; Zhai et al.,

2011; Guo et al., 2010), they all assume that aspect terms have been extracted beforehand.

In recent years, topic models have been used to perform extraction and grouping at the same time. Existing works are based on two basic models, pLSA (Hofmann, 1999) and LDA (Blei et al., 2003). Some existing works include discovering global and local aspects (Titov and McDonald, 2008), extracting key phrases (Branavan et al., 2008), rating multi-aspects (Wang et al., 2010; Moghaddam and Ester, 2011), summarizing aspects and sentiments (Lu et al., 2009), and modeling attitudes (Sauper et al., 2011). In (Lu and Zhai, 2008), a semi-supervised model was proposed. However, their method is entirely different from ours as they use expert reviews to guide the analysis of user reviews.

Aspect and sentiment extraction using topic modeling come in two flavors: discovering aspect words sentiment wise (i.e., discovering positive and negative aspect words and/or sentiments for each aspect without separating aspect and sentiment terms) (Lin and He, 2009; Brody and Elhadad, 2010; Jo and Oh, 2011) and separately discovering both aspects and sentiments (e.g., Mei et al., 2007; Zhao et al., 2010). Zhao et al. (2010) used Maximum-Entropy to train a switch variable to separate aspect and sentiment words. We adopt this method as well but with no use of manually labeled data in training. One problem with these existing models is that many discovered aspects are not understandable/meaningful to users. Chang et al. (2009) stated that one reason is that the objective function of topic models does not always correlate well with human judgments. Our seeded models are designed to overcome this problem.

Researchers have tried to generate “meaningful” and “specific” topics/aspects. Blei and McAuliffe (2007) and Ramage et al. (2009) used document label information in a supervised setting. Hu et al. (2011) relied on user feedback during Gibbs sampling iterations. Andrzejewski et al. (2011) incorporated first-order logic with Markov Logic Networks. However, it has a practical limitation for reasonably large corpora since the number of non-trivial groundings can grow to  $O(N^2)$  where  $N$  is the number of unique tokens in the corpus. Andrzejewski et al. (2009) used another approach (DF-LDA) by introducing must-link and cannot-link constraints as Dirichlet Forest priors. Zhai et al. (2011) reported that the model does not scale up



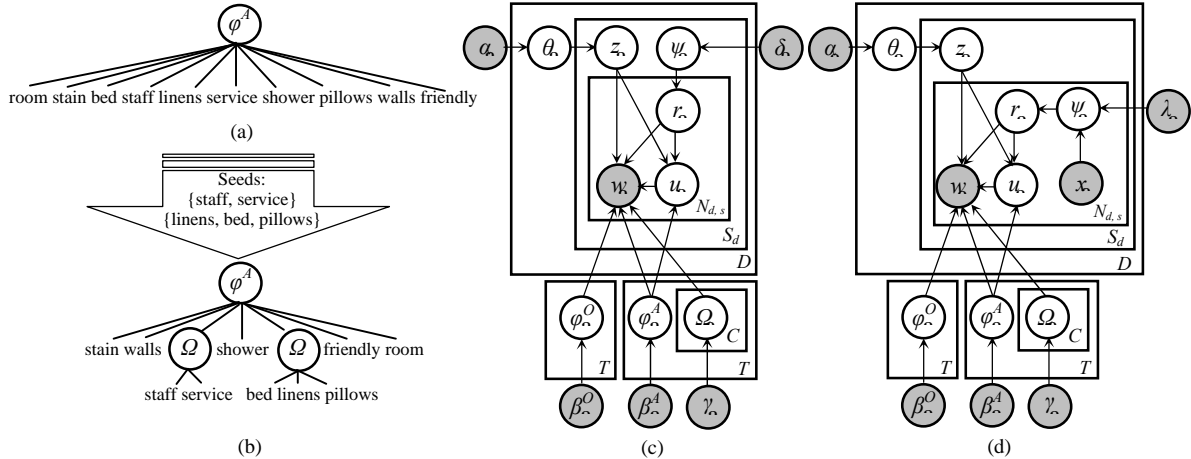


Figure 1: Prior structure: (a) Standard ASMs, (b) Two-level tree structured distribution. Graphical models in plate notation: (c) SAS and (d) ME-SAS.

when the number of cannot-links go beyond 1000 because the number of maximal cliques  $Q^{(r)}$  in a connected component of size  $|r|$  in the cannot-link graph is exponential in  $r$ . Note that we could still experiment with DF-LDA as our problem size is not so large. We will show in Section 4 that the proposed models outperform it by a large margin.

### 3 Proposed Seeded Models

The standard LDA and existing aspect and sentiment models (ASMs) are mostly governed by the phenomenon called “higher-order co-occurrence” (Heinrich, 2009), i.e., based on how often terms co-occur in different contexts<sup>1</sup>. This unfortunately results in many “non-specific” terms being pulled and clustered. We employ seed sets to address this issue by “guiding” the model to group semantically related terms in the same aspect thus making the aspect more specific and related to the seeds (which reflect the user needs). For easy presentation, we will use *aspect* to mean *aspect category* from now on. We replace the multinomial distribution over words for each aspect (as in ASMs) with a special two-level tree structured distribution. The generative process of ASMs assumes that each vocabulary word is independently (i.e., not dependent upon other word-aspect association) and equally probable to be associated with any aspect. Due to higher-order co-occurrences, we find conceptually different terms yet related in contexts (e.g., in hotel domain terms like stain, shower, walls in aspect

*Maintenance*; bed, linens, pillows in aspect *Cleanliness*) equally probable of emission for any aspect. Figure 1(a) shows an example tree. Upon adding the seed sets {bed, linens, pillows} and {staff, service}, the prior structure now changes to the correlated distribution in Figure 1 (b). Thus, each aspect has a top level distribution over non-seed words and seed sets. Each seed set in each aspect further has a second level distribution over seeds in that seed set. The aspect term (word) emission now requires two steps: first sampling at level one to obtain a non-seed word or a seed set. If a non-seed word is sampled we emit it else we further sample at the second seed set level and emit a seed word. This ensures that seed words together have either all high or low aspect associations. Furthermore, seed sets preserve conjugacy between related concepts and also shape more specific aspects by clustering based on higher order co-occurrences with seeds rather than only with standard one level multinomial distribution over words (or terms) alone.

#### 3.1 SAS Model

We now present the proposed Seeded Aspect and Sentiment model (SAS). Let  $v_{1..V}$  denote the entries in our vocabulary where  $V$  is the number of unique non-seed terms. Let there be  $C$  seed sets  $Q_{l=1..C}$  where each seed set  $Q_l$  is a group of semantically related terms. Let  $\varphi_{t=1..T}^A, \varphi_{t=1..T}^O$  denote  $T$  aspect and aspect specific sentiment models. Also let  $\Omega_{t,l}$  denote the aspect specific distribution of seeds in the seed set  $Q_l$ . Following the approach of (Zhao et al., 2010), we too assume that a review sentence usually talks about one

<sup>1</sup>  $w_1$  co-occurring with  $w_2$  which in turn co-occurs with  $w_3$  denotes a second-order co-occurrence between  $w_1$  and  $w_3$ .

aspect. A review document  $d_{1...D}$  comprises of  $S_d$  sentences and each sentence  $s \in S_d$  has  $N_{d,s}$  words. Also, let  $Sent_s^d$  denote the sentence  $s$  of document  $d$ . To distinguish between aspect and sentiment terms, we introduce an indicator (switch) variable  $r_{d,s,j} \in \{\hat{a}, \hat{o}\}$  for the  $j^{th}$  term of  $Sent_s^d, w_{d,s,j}$ . Further, let  $\psi_{d,s}$  denote the distribution of aspects and sentiments in  $Sent_s^d$ . The generative process of the SAS model (see Figure 1(c)) is given by:

1. For each aspect  $t \in \{1, \dots, T\}$ :
  - i. Draw  $\varphi_t^o \sim Dir(\beta^o)$
  - ii. Draw a distribution over terms and seed sets  $\varphi_t^A \sim Dir(\beta^A)$ 
    - a) For each seed set  $l \in \{Q_1, \dots, Q_C\}$   
Draw a distribution over seeds  $\Omega_{t,l} \sim Dir(\gamma)$
2. For each (review) document  $d \in \{1, \dots, D\}$ :
  - i. Draw  $\theta_d \sim Dir(\alpha)$
  - ii. For each sentence  $s \in \{1, \dots, S_d\}$ :
    - a) Draw  $z_{d,s} \sim Mult(\theta_d)$
    - b) Draw  $\psi_{d,s} \sim Beta(\delta)$
    - c) For each term  $w_{d,s,j}$  where  $j \in \{1, \dots, N_{d,s}\}$ :
      - I. Draw  $r_{d,s,j} \sim Bernoulli(\psi_{d,s})$ ,  $r_{d,s,j} \in \{\hat{a}, \hat{o}\}$
      - II. if  $r_{d,s,j} = \hat{o}$  //  $w_{d,s,j}$  is a sentiment  
Emit  $w_{d,s,j} \sim Mult(\varphi_{z_{d,s}}^o)$   
else //  $r_{d,s,j} = \hat{a}$ ,  $w_{d,s,j}$  is an aspect
        - A. Draw  $u_{d,s,j} \sim Mult(\varphi_{z_{d,s}}^A)$
        - B. if  $u_{d,s,j} \in V$  // non-seed term  
Emit  $w_{d,s,j} = u_{d,s,j}$   
else //  $u_{d,s,j}$  is some seed set index say  $l_{d,s,j}$   
Emit  $w_{d,s,j} \sim \Omega_{z_{d,s}, l_{d,s,j}}$

We employ collapsed Gibbs sampling (Griffiths and Steyvers, 2004) for posterior inference. As  $z$  and  $r$  are at different hierarchical levels, we derive their samplers separately as follows:

$$p(z_{d,s} = t | Z_{-d,s}, R_{-d,s}, W_{-d,s}, U_{-d,s}) \propto \frac{B(n_{t,[]-d,s}^o + \beta^o)}{B(n_{t,[]-d,s}^o + \beta^o)} \times \frac{B(n_{t,[]-d,s}^{U,A} + \beta^A)}{B(n_{t,[]-d,s}^{U,A} + \beta^A)} \times \prod_{l=1}^C \frac{B(n_{t,l,[]-d,s}^{S,A} + \gamma)}{B(n_{t,l,[]-d,s}^{S,A} + \gamma)} \times \frac{n_{d,t}^{Sent. -d,s} + \alpha}{n_{d,(.)}^{Sent. -d,s} + T\alpha} \quad (1)$$

$$p(r_{d,s,j} = \hat{o} | Z_{-d,s}, R_{-d,s,j}, W_{-d,s,j}, U_{-d,s,j}, z_{d,s} = t, w_{d,s,j} = w) \propto \frac{n_{t,w-d,s,j}^o + \beta^o}{n_{t,(.)-d,s,j}^o + |V \cup U_l Q_l| \beta^o} \times \frac{n_{d,s-d,s,j}^o + \delta_b}{n_{d,s-d,s,j}^o + \delta_a + n_{d,s-d,s,j}^o + \delta_b} \quad (2)$$

$$p(r_{d,s,j} = \hat{a} | \dots) \propto \begin{cases} \frac{n_{t,l,w-d,s,j}^{S,A} + \gamma}{n_{t,l,(.)-d,s,j}^{S,A} + |Q_l| \gamma} \times \frac{n_{t,l}^o + \beta^A}{n_{t,(.)}^o + (V+C)\beta^A} \times \frac{n_{d,s-d,s,j}^o + \delta_b}{n_{d,s-d,s,j}^o + \delta_a + n_{d,s-d,s,j}^o + \delta_b} ; w \in Q_l \\ \frac{n_{t,w}^{U,A} + \beta^A}{n_{t,(.)}^{U,A} + (V+C)\beta^A} \times \frac{n_{d,s-d,s,j}^o + \delta_b}{n_{d,s-d,s,j}^o + \delta_a + n_{d,s-d,s,j}^o + \delta_b} ; \forall l, w \in Q_l \end{cases} \quad (3)$$

where  $B(\vec{x}) = \frac{\prod_{i=1}^{\dim(\vec{x})} \Gamma(x_i)}{\Gamma(\sum_{i=1}^{\dim(\vec{x})} x_i)}$  is the multinomial Beta function.  $n_{t,v}^o$  is the number of times term  $v$  was

assigned to aspect  $t$  as an opinion/sentiment word.  $n_{t,v}^{U,A}$  is the number of times non-seed term  $v \in V$  was assigned to aspect  $t$  as an aspect.  $n_{t,v}^{S,A}$  is the number of times seed term  $v \in V_l$  was assigned to aspect  $t$  as an aspect.  $n_{d,t}^{Sent.}$  is the number of sentences in document  $d$  that were assigned to aspect  $t$ .  $n_{d,s}^A$  and  $n_{d,s}^o$  denote the number of terms in  $Sent_s^d$  that were assigned to aspects and opinions respectively.  $n_{t,l}^o$  is the number of times any term of seed set  $Q_l$  was assigned to aspect  $t$ . Omission of a latter index denoted by  $[]$  in the above notation represents the corresponding row vector spanning over the latter index. For example,  $n_{t,[]}^{U,A} = [n_{t,v=1}^{U,A}, \dots, n_{t,v=V}^{U,A}]$  and  $(\cdot)$  denotes the marginalized sum over the latter index. The subscript  $-d,s$  denotes the counts excluding assignments of all terms in  $Sent_s^d$ .  $-d,s,j$  denotes counts excluding  $w_{d,s,j}$ . We perform hierarchical sampling. First, an aspect is sampled for each sentence  $z_{d,s}$  using Eq. (1). After sampling the aspect, we sample  $r_{d,s,j}$ . The probability of  $w_{d,s,j}$  being an opinion or sentiment term,  $p(r_{d,s,j} = \hat{o})$  is given by Eq. (2). However, for  $p(r_{d,s,j} = \hat{a})$  we have two cases: (a) the observed term  $w = w_{d,s,j} \in Q_l$  or (b) does not belong to any seed set,  $\forall l, w \in Q_l$ , i.e.,  $w$  is a non-seed term. These cases are dealt in Eq. (3).

**Asymmetric Beta priors:** Hyper-parameters  $\alpha, \beta^o, \beta^A$  are not very sensitive and the heuristic values suggested in (Griffiths and Steyvers, 2004) usually hold well in practice (Wallach et al. 2009). However, the smoothing hyper-parameter  $\delta$  (Figure 1(c)) is crucial as it governs the aspect or sentiment switch. Essentially,  $\psi_{d,s} \sim Beta(\delta \vec{\xi})$  is the probability of emitting an aspect term<sup>2</sup> in  $Sent_s^d$  with concentration parameter  $\delta$  and base measure  $\vec{\xi} = [\xi_a, \xi_b]$ . Without any prior belief, uniform base measures  $\xi_a = \xi_b = 0.5$  are used resulting in symmetric Beta priors. However, aspects are often more probable than sentiments in a sentence (e.g., “The beds, sheets, and bedding were dirty.”). Thus, it is more principled to employ asymmetric priors. Using a labeled set of sentences,  $S_{labeled}$ , where we know the per sentence probability of aspect emission ( $\psi_{d,s}$ ), we can employ the method of moments to estimate the smoothing hyper-parameter  $\delta = [\delta_a, \delta_b]$ :

$$\delta_a = \mu \left( \frac{\mu(1-\mu)}{\sigma} - 1 \right), \delta_b = \delta_a \left( \frac{1}{\mu} - 1 \right); \mu = E[\psi_{d,s}], \sigma = Var[\psi_{d,s}] \quad (4)$$

<sup>2</sup>  $r_{d,s,j} \sim Bernoulli(\psi_{d,s})$ .  $\psi_{d,s}, 1 - \psi_{d,s}$  are the success and failure probability of emitting an aspect/sentiment term.

### 3.2 ME-SAS Model

We can further improve SAS by employing Maximum Entropy (Max-Ent) priors for aspect and sentiment switching. We call this new model ME-SAS. The motivation is that aspect and sentiment terms play different syntactic roles in a sentence. Aspect terms tend to be nouns or noun phrases while sentiment terms tend to be adjectives, adverbs, etc. POS tag information can be elegantly encoded by moving  $\psi_{d,s}$  to the term plate (see Figure 1(d)) and drawing it from a Max-Ent( $x_{d,s,j}; \lambda$ ) model. Let  $\vec{x}_{d,s,j} = [POS_{w_{d,s,j-1}}, POS_{w_{d,s,j}}, POS_{w_{d,s,j+1}}, w_{d,s,j} - 1, w_{d,s,j}, w_{d,s,j} + 1]$  denote the feature vector associated with  $w_{d,s,j}$  encoding lexical and POS features of the previous, current and next term. Using a training data set, we can learn Max-Ent priors. Note that unlike traditional Max-Ent training, we do not need manually labeled data for training (see Section 4 for details). For ME-SAS, only the sampler for the switch variable  $r$  changes as follows:

$$p(r_{d,s,j} = \hat{o} | Z_{-d,s}, R_{-d,s,j}, W_{-d,s,j}, U_{-d,s,j}, Z_{d,s} = t, w_{d,s,j} = w) \propto \frac{n_{t,w_{-d,s,j}}^{\hat{o}} + \beta^{\hat{o}}}{n_{t,(c)-d,s,j}^{\hat{o}} + |V \cup U \cup Q_l| \beta^{\hat{o}}} \times \frac{\exp(\sum_{i=1}^n \lambda_{if_i}(x_{d,s,j}, \hat{o}))}{\sum_{y \in \{\hat{a}, \hat{o}\}} \exp(\sum_{i=1}^n \lambda_{if_i}(x_{d,s,j}, y))} \quad (5)$$

$$p(r_{d,s,j} = \hat{a} | \dots) \propto \begin{cases} \frac{n_{t,w_{-d,s,j}}^{S,A} + \gamma}{n_{t,(c)-d,s,j}^{S,A} + |Q_l| \gamma} \times \frac{n_{t,l}^{\hat{a}} + \beta^{\hat{a}}}{n_{t,(c)}^{\hat{a}} + (V+C)\beta^{\hat{a}}} \times \frac{\exp(\sum_{i=1}^n \lambda_{if_i}(x_{d,s,j}, \hat{a}))}{\sum_{y \in \{\hat{a}, \hat{o}\}} \exp(\sum_{i=1}^n \lambda_{if_i}(x_{d,s,j}, y))} ; w \in Q_l \\ \frac{n_{t,w}^{U,A} + \beta^{\hat{a}}}{n_{t,(c)}^{U,A} + (V+C)\beta^{\hat{a}}} \times \frac{\exp(\sum_{i=1}^n \lambda_{if_i}(x_{d,s,j}, \hat{a}))}{\sum_{y \in \{\hat{a}, \hat{o}\}} \exp(\sum_{i=1}^n \lambda_{if_i}(x_{d,s,j}, y))} ; \hat{a} \in l, w \in Q_l \end{cases} \quad (6)$$

where  $\lambda_{1..n}$  are the parameters of the learned Max-Ent model corresponding to the  $n$  binary feature functions  $f_{1..n}$  of Max-Ent.

## 4 Experiments

This section evaluates the proposed models. Since the focus in this paper is to generate high quality aspects using seeds, we will not evaluate sentiments although both SAS and ME-SAS can also discover sentiments. To compare the performance with our models, we use two existing state-of-the-art models, ME-LDA (Zhao et al. 2010) and DF-LDA (Andrzejewski et al., 2009). As discussed in Section 2, there are two main flavors of aspect and sentiment models. The first flavor does not separate aspect and sentiment, and the second flavor uses a switch to perform the separation. Since our models also perform a

switch, it is natural to compare with the latter flavor, which is also more advanced. ME-LDA is the representative model in this flavor. DF-LDA adds constraints to LDA. We use our seeds to generate constraints for DF-LDA. While ME-LDA cannot consider constraints, DF-LDA does not separate sentiments and aspects. Apart from other modeling differences, our models can do both, which enable them to produce much better results.

**Dataset and Settings:** We used hotel reviews from tripadvisor.com. Our corpus consisted of 101,234 reviews and 692,783 sentences. Punctuations, stop words<sup>3</sup>, and words appearing less than 5 times in the corpus were removed.

For all models, the posterior inference was drawn after 5000 Gibbs iterations with an initial burn-in of 1000 iterations. For SAS and ME-SAS, we set  $\alpha = 50/T$ ,  $\beta^A = \beta^O = 0.1$  as suggested in (Griffiths and Steyvers, 2004). To make the seeds more effective, we set the seed set word-distribution hyper-parameter  $\gamma$  to be much larger than  $\beta^A$ , the hyper-parameter for the distribution over seed sets and aspect terms. This results in higher weights to seeded words which in turn guide the sampler to cluster relevant terms better. A more theoretical approach would involve performing hyper-parameter estimation (Wallach et al., 2009) which may reveal specific properties of the dataset like the estimate of  $\alpha$  (indicating how different documents are in terms of their latent semantics),  $\beta$  (suggesting how large the groups of frequently appearing aspect and sentiment terms are) and  $\gamma$  (giving a sense of which and how large groupings of seeds are good). These are interesting questions and we defer it to our future work. In this work, we found that the setting  $\gamma = 250$ , a larger value compared to  $\beta^A$ , produced good results.

For SAS, the asymmetric Beta priors were estimated using the method of moments (Section 3.1). We sampled 500 random sentences from the corpus and for each sentence identified the aspects. We thus computed the per-sentence probability of aspect emission ( $\psi_{d,s}$ ) and used Eq. (4) to compute the final estimates, which give  $\delta_a = 2.35$ ,  $\delta_b = 3.44$ .

To learn the Max-Ent parameters  $\lambda$  of ME-SAS, we used the sentiment lexicon<sup>4</sup> of (Hu and Liu, 2004) to automatically generate training data (no manual labeling). We randomly sampled 1000 terms from the corpus which have appeared at least

<sup>3</sup> <http://jmlr.csail.mit.edu/papers/volume5/Iewis04a/a11-smart-stop-list/english.stop>

<sup>4</sup> <http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar>

Aspect (seeds)	ME-SAS		SAS		ME-LDA		DF-LDA
	Aspect	Sentiment	Aspect	Sentiment	Aspect	Sentiment	Topic
<b>Staff</b> (staff service waiter hospitality upkeep)	attendant manager waitress maintenance bartender waiters housekeeping receptionist waitstaff janitor	friendly attentive polite nice <b>clean</b> pleasant <b>slow</b> courteous rude professional	attendant waiter waitress manager maintenance waiters housekeeping receptionist <b>polite</b>	friendly nice dirty <b>comfortable</b> nice <b>clean</b> polite <b>extremely</b> courteous efficient	staff maintenance <b>room</b> upkeep <b>linens</b> room-service receptionist <b>wait</b> <b>pillow</b> waiters	friendly nice courteous <b>extremely</b> nice <b>clean</b> polite little helpful <b>better</b>	staff <b>friendly</b> <b>helpful</b> <b>beds</b> <b>front</b> room <b>comfortable</b> <b>large</b> receptionist housekeeping
<b>Cleanliness</b> (curtains restroom floor beds cleanliness)	carpets hall towels bathtub couch mattress linens wardrobe spa pillow	clean dirty comfortable fresh wet filthy extra stain <b>front</b> worn	hall carpets towels pillow stain mattress <b>filthy</b> linens <b>interior</b> bathtub	clean dirty fresh old nice good <b>enough</b> new <b>front</b> <b>friendly</b>	cleanliness floor carpets bed <b>lobby</b> bathroom <b>staff</b> closet spa <b>décor</b>	clean good dirty <b>hot</b> <b>large</b> nice <b>fresh</b> <b>thin</b> new <b>little</b>	clean pool <b>beach</b> carpets <b>parking</b> bed bathroom <b>nice</b> comfortable suite
<b>Comfort</b> (comfort mattress furniture couch pillows)	bedding bedcover sofa linens bedroom suites <b>décor</b> comforter blanket futon	comfortable clean soft nice uncomfortable spacious hard comfy dirty quiet	bed linens sofa bedcover <b>hard</b> bedroom <b>privacy</b> <b>double</b> <b>comfy</b> futon	nice dirty comfortable large clean <b>best</b> spacious <b>only</b> big <b>extra</b>	bed mattress suites furniture <b>lighting</b> <b>décor</b> room bedroom <b>hallway</b> <b>carpet</b>	great clean awesome dirty best comfortable soft nice <b>only</b> <b>extra</b>	bed mattress <b>nice</b> <b>stay</b> <b>lighting</b> lobby comfort room <b>dirty</b> sofa

Table 1: Top ranked aspect and sentiment words in three aspects (please see the explanation in Section 4.1).

20 times (to ensure that the training set is reasonably representative of the corpus). Of those 1000 terms if they appeared in the sentiment lexicon, they were treated as sentiment terms, else aspect terms. Clearly, labeling words not in the sentiment lexicon as aspect terms may not always be correct. Even with this noisy automatically-labeled data, the proposed models can produce good results. Since ME-LDA used manually labeled training data for Max-Ent, we again randomly sampled 1000 terms from our corpus appearing at least 20 times and labeled them as aspect terms or sentiment terms, so this labeled data clearly has less noise than our automatically labeled data. For both ME-SAS and ME-LDA we used the corresponding feature vector of each labeled term (in the context of sentences where it occurs) to train the Max-Ent model. As DF-LDA requires must-link and cannot-link constraints, we used our seed sets to generate intra-seed set must-link and inter-seed set cannot-link constraints. For its hyper-parameters, we used the default values in the package<sup>5</sup> (Andrzejewski et al., 2009).

Setting the number of topics/aspects in topic models is often tricky as it is difficult to know the

exact number of topics that a corpus has. While non-parametric Bayesian approaches (Teh et al., 2006) do exist for estimating the number of topics,  $T$ , they strongly depend on the hyper-parameters (Heinrich, 2009). As we use fixed hyper-parameters, we do not learn  $T$  from Bayesian non-parametrics. We used 9 major aspects ( $T = 9$ ) based on commonsense knowledge of what people usually talk about hotels and some experiments. These are *Dining*, *Staff*, *Maintenance*, *Check In*, *Cleanliness*, *Comfort*, *Amenities*, *Location* and *Value for Money* (VFM). However, it is important to note that the proposed models are flexible and do not need to have seeds for every aspect/topic. Our experiments simulate the real-life situation where the user may not know all aspects or have no seeds for some aspects. Thus, we provided seeds only to the first 6 of the 9 aspects/topics. We will see that without seeds for all aspects, our models not only can improve the seeded aspects but also improve the non-seeded aspects.

#### 4.1 Qualitative Results

This section shows some qualitative results to give an intuitive feeling of the results from different models. Table 1 shows the aspect terms and sentiment terms discovered by the 4 models for

<sup>5</sup> [http://pages.cs.wisc.edu/~andrzej/research/df\\_lda.html](http://pages.cs.wisc.edu/~andrzej/research/df_lda.html)

three aspects. Due to space limitations, we are unable to show all 6 aspects for which we have seeds. Since DF-LDA cannot separate aspects and sentiments, we only show its topics (aspects). **Red (bold)** colored words show semantic clustering errors or inappropriate terms for different groups.

It is important to note that we judge the results based on how they are related to the user seeds (which represent the user need). The judgment is to some extent subjective. What we reported here are based on our judgments what are appropriate and what are not for each aspect. For SAS, ME-SAS and ME-LDA, we mark sentiment terms as errors when they are grouped under aspects as these models are supposed to separate sentiments and aspects. For DF-LDA, the situation is different as it is not meant to separate sentiment and aspect terms, we use *red* italic font to indicate those adjectives which are aspect specific adjectives (see more discussion below). Our judgment may be slightly unfair to ME-LDA and DF-LDA as their results may make sense in some other ways. However, that is precisely the purpose of this work, to produce results that suit the user’s need rather than something generic.

We can see from Table 1 that ME-SAS performs the best. Next in order are SAS, ME-LDA, and DF-LDA. We see that only providing a handful of seeds (5) for the aspect *Staff*, ME-SAS can discover highly specific words like manager, attendant, bartender, and janitor. By specific, we mean they are highly related to the given seeds. While SAS also discovers specific words benefiting from seeds, relying on Beta priors for aspect and sentiment switching was less effective. Next in performance is ME-LDA which although produces reasonable results in general, several aspect terms are far from what the user wants based on the seeds, e.g., room, linens, wait, pillow. Finally, we observe that DF-LDA does not perform well either. One reason is that it is unable to separate aspects and sentiments. Although encoding the intra-seed set must-link and inter-seed set cannot-link constraints in DF-LDA discovers some specific words as ME-SAS, they are much lower in the ranked order and hence do not show up in the top 10 words in Table 1. As DF-LDA is not meant to perform extraction and to group both aspect and sentiment terms, we relax the errors of DF-LDA due to correct aspect specific sentiments (e.g., friendly, helpful for *Staff* are correct aspect specific sentiments, but still

regard incorrect sentiments like front, comfortable, large as errors) placed in aspect models. We call this model DF-LDA-Relaxed.

## 4.2 Quantitative Results

Topic models are often evaluated quantitatively using perplexity and likelihood on held-out test data (Blei et al., 2003). However, perplexity does not reflect our purpose since our aim is not to predict whether an unseen document is likely to be a review of some particular aspect. Nor are we trying to evaluate how well the unseen review data fits our seeded models. Instead our focus is to evaluate how well our learned aspects perform in clustering specific terms guided by seeds. So we directly evaluate the discovered aspect terms. Note again we do not evaluate sentiment terms as they are not the focus of this paper<sup>6</sup>. Since aspects produced by the models are rankings and we do not know the number of correct aspect terms, a natural way to evaluate these rankings is to use *precision @ n* (or  $p@n$ ), where  $n$  is a rank position.

**Varying number of seeds:** Instead of a fixed number of seeds, we want to see the effect of the number of seeds on aspect discovery. Table 2 reports the average  $p@n$  vs. the number of seeds. The average is a two-way averaging. The first average was taken over all combinations of actual seeds selected for each aspect, e.g., when the number of seeds is 3, out of the 5 seeds in each aspect, all  $\binom{5}{3}$  combinations of seeds were tried and the results averaged. The results were further averaged over  $p@n$  for 6 aspects with seeds. We start with 2 seeds and progressively increase them to 5. Using only 1 seed per seed set (or per aspect) has practically no effect because the top level distribution  $\varphi^A$  encodes which seed sets (and non-seed words) to include; the lower-level distribution  $\mathcal{Q}$  constrains the probabilities of the seed words to be correlated for each of the seed sets. Thus, having only one seed per seed set will result in sampling that single word whenever that seed set is chosen which will not have the effect of correlating seed words so as to pull other words based on co-occurrence with constrained seed words. From Table 2, we can see that for all models  $p@n$  progressively improves as the number of seeds increases. Again ME-SAS performs the best followed by SAS and DF-LDA.

<sup>6</sup> A qualitative evaluation of sentiment extraction based on Table 1 yields the following order: ME-SAS, SAS, ME-LDA.

No. of Seeds	DF-LDA			DF-LDA-Relaxed			SAS			ME-SAS		
	P@10	P@20	P@30	P@10	P@20	P@30	P@10	P@20	P@30	P@10	P@20	P@30
2	0.51	0.53	0.49	0.67	0.69	0.70	0.69	0.71	0.67	0.74	0.72	0.70
3	0.53	0.54	0.50	0.71	0.70	0.71	0.71	0.72	0.70	0.78	0.75	0.72
4	0.57	0.56	0.53	0.73	0.73	0.73	0.75	0.74	0.73	0.83	0.79	0.76
5	0.59	0.57	0.54	0.75	0.74	0.75	0.77	0.76	0.74	0.86	0.81	0.77

Table 2: Average  $p@n$  of the seeded aspects with the no. of seeds.

Aspect	ME-LDA			DF-LDA			DF-LDA-Relaxed			SAS			ME-SAS		
	P@10	P@20	P@30	P@10	P@20	P@30	P@10	P@20	P@30	P@10	P@20	P@30	P@10	P@20	P@30
Dining	0.70	0.65	0.67	0.50	0.60	0.63	0.70	0.70	0.70	0.80	0.75	0.73	0.90	0.85	0.80
Staff	0.60	0.70	0.67	0.40	0.65	0.60	0.60	0.75	0.67	0.80	0.80	0.70	1.00	0.90	0.77
Maintenance	0.80	0.75	0.73	0.40	0.55	0.56	0.60	0.70	0.73	0.70	0.75	0.76	0.90	0.85	0.80
Check In	0.70	0.70	0.67	0.50	0.65	0.60	0.80	0.75	0.70	0.80	0.70	0.73	0.90	0.80	0.76
Cleanliness	0.70	0.75	0.67	0.70	0.70	0.63	0.70	0.75	0.70	0.80	0.75	0.70	1.00	0.85	0.83
Comfort	0.60	0.70	0.63	0.60	0.65	0.50	0.70	0.75	0.63	0.60	0.75	0.67	0.90	0.80	0.73
Amenities	0.80	0.80	0.67	0.70	0.65	0.53	0.90	0.75	0.73	0.90	0.80	0.70	1.00	0.85	0.73
Location	0.60	0.70	0.63	0.50	0.60	0.56	0.70	0.70	0.67	0.60	0.70	0.63	0.70	0.75	0.67
VFM	0.50	0.55	0.50	0.40	0.50	0.46	0.60	0.60	0.60	0.50	0.50	0.50	0.60	0.55	0.53
Avg.	0.67	0.70	0.65	0.52	0.62	0.56	0.70	0.72	0.68	0.72	0.72	0.68	0.88	0.80	0.74

Table 3: Effect of performance on seeded and non-seeded aspects (5 seeds were used for the 6 seeded aspects).

**Effect of seeds on non-seeded aspects:** Here we compare all models aspect wise and see the results of seeded models SAS and ME-SAS on non-seeded aspects (Table 3). Shaded cells in Table 3 give the  $p@n$  values for DF-LDA, DF-LDA-Relaxed, SAS, and ME-SAS on three non-seeded aspects (Amenities, Location, and VFM)<sup>7</sup>.

We see that across all the first 6 aspects with (5) seeds ME-SAS outperforms all other models by large margins in all top 3 ranked buckets  $p@10$ ,  $p@20$  and  $p@30$ . Next in order are SAS, ME-LDA and DF-LDA. For the last three aspects which did not have any seed guidance, we find something interesting. Seeded models SAS and especially ME-SAS result in improvements of non-seeded aspects too. This is because as seeds facilitate clustering specific and appropriate terms in seeded aspects, which in turn improves precision on non-seeded aspects. This phenomenon can be clearly seen in Table 1. In aspect *Staff* of ME-LDA, we find *pillow* and *linens* being clustered. This is not a “flaw” of the model per se, but the point here is *pillow* and *linens* happen to co-occur many times with other words like *maintenance*, *staff*, and *upkeep* because “room-service” generally includes staff members coming and replacing linens and pillow covers. Although *pillow* and *linens* are related to *Staff*, strictly speaking they are semantically incorrect because they do not represent the very concept “Staff” based on the seeds (which reflect the user need). Presence of

seed sets in SAS and ME-SAS result in pulling such words as linens and pillow (due to seeds like beds and cleanliness in the aspect *Cleanliness*) and ranking them higher in the aspect *Cleanliness* (see Table 1) where they make more sense than *Staff*. Lastly, we also note that the improvements in non-seeded aspects are more pronounced for ME-SAS than SAS as SAS encounters more switching errors which counters the improvement gained by seeds.

In summary, the averages over all aspects (Table 3 last row) show that the proposed seeded models SAS and ME-SAS outperform ME-LDA, DF-LDA and even DF-LDA-Relaxed considerably.

## 5 Conclusion

This paper studied the issue of using seeds to discover aspects in an opinion corpus. To our knowledge, no existing work deals with this problem. Yet, it is important because in practice the user often has something in mind to find. The results obtained in a completely unsupervised manner may not suit the user’s need. To solve this problem, we proposed two models SAS and ME-SAS which take seeds reflecting the user needs to discover specific aspects. ME-SAS also does not need any additional help from the user in its Max-Ent training. Our results showed that both models outperformed two state-of-the-art existing models ME-LDA and DF-LDA by large margins.

## Acknowledgments

This work is supported in part by National Science Foundation (NSF) under grant no. IIS-1111092.

<sup>7</sup> Note that Tables 2 and 3 are different runs of the model. The variations in the results are due to the random initialization of the Gibbs sampler.

## References

- Andrzejewski, D., Zhu, X. and Craven, M. 2009. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. Proceedings of International Conference on Machine Learning (ICML).
- Andrzejewski, D., Zhu, X. and Craven, M. and Recht, B. 2011. A framework for incorporating general domain knowledge into latent Dirichlet allocation using first-order logic. Proceedings of the 22nd International Joint Conferences on Artificial Intelligence (IJCAI).
- Blair-Goldensohn, S., Hannan, K., McDonald, R., Neylon, T., Reis, G. A. and Reynar, J. 2008. Building a sentiment summarizer for local service reviews. Proceedings of WWW-2008 workshop on NLP in the Information Explosion Era.
- Blei, D., Ng, A. and Jordan, M. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research* 3: 993-1022.
- Blei D. and McAuliffe, J. 2007. Supervised topic models. *Neural Information Processing Systems (NIPS)*.
- Branavan, S., Chen, H., Eisenstein J. and Barzilay, R. 2008. Learning document-level semantic properties from free-text annotations. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL).
- Brody, S. and Elhadad, S. 2010. An Unsupervised Aspect-Sentiment Model for Online Reviews. Proceedings of The 2010 Annual Conference of the North American Chapter of the ACL (NAACL).
- Carenini, G., Ng, R. and Zwart, E. 2005. Extracting knowledge from evaluative text. Proceedings of Third Intl. Conf. on Knowledge Capture (K-CAP-05).
- Chang, J., Boyd-Graber, J., Wang, C. Gerrish, S. and Blei, D. 2009. Reading tea leaves: How humans interpret topic models. In *Neural Information Processing Systems (NIPS)*.
- Choi, Y. and Cardie, C. 2010. Hierarchical sequential learning for extracting opinions and their attributes. Proceedings of Annual Meeting of the Association for Computational (ACL).
- Griffiths, T. and Steyvers, M. 2004. Finding scientific topics. *Proceedings of National Academy of Sciences (PNAS)*.
- Guo, H., Zhu, H., Guo, Z., Zhang, X. and Su, X. 2009. Product feature categorization with multilevel latent semantic association. Proceedings of ACM International Conference on Information and Knowledge Management (CIKM).
- Heinrich, G. 2009. A Generic Approach to Topic Models. Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD).
- Hofmann, T. 1999. Probabilistic latent semantic indexing. Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI).
- Hu, Y., Boyd-Graber, J. and Satinoff, B. 2011. Interactive topic modeling. Proceedings of Annual Meeting of the Association for Computational Linguistics (ACL), 2011.
- Hu, M. and Liu, B. 2004. Mining and summarizing customer reviews. *International Conference on Knowledge Discovery and Data Mining (ICDM)*.
- Jakob, N. and Gurevych, I. 2010. Extracting Opinion Targets in a Single-and Cross-Domain Setting with Conditional Random Fields. Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP).
- Jin, W. and Ho, H. 2009. A novel lexicalized HMM-based learning framework for web opinion mining. Proceedings of International Conference on Machine Learning (ICML).
- Jo, Y. and Oh, A. 2011. Aspect and sentiment unification model for online review analysis. *ACM Conference in Web Search and Data Mining (WSDM)*.
- Kobayashi, N., Inui, K. and Matsumoto, K. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL).
- Ku, L., Liang, Y. and Chen, H. 2006. Opinion extraction, summarization and tracking in news and blog corpora. Proceedings of AAAI Symposium on Computational Approaches to Analyzing Weblogs (AAAI-CAAW'06).
- Li, F., Han, C., Huang, M., Zhu, X. Xia, Y., Zhang, S. and Yu, H. 2010. Structure-aware review mining and summarization. *International Conference on Computational Linguistics (COLING)*.
- Lin, C. and He, Y. 2009. Joint sentiment/topic model for sentiment analysis. Proceedings of ACM International Conference on Information and Knowledge Management (CIKM).
- Liu, B. 2012. Sentiment Analysis and Opinion Mining.



- Morgan & Claypool publishers (to appear in June 2012).
- Liu, B., M. Hu, and J. Cheng. 2005. Opinion Observer: Analyzing and comparing opinions on the web. Proceedings of International Conference on World Wide Web (WWW).
- Lu, Y., Zhai, C. and Sundaresan, N. 2009. Rated aspect summarization of short comments. Proceedings of International Conference on World Wide Web (WWW).
- Lu, Y. and Zhai, C. 2008. Opinion Integration Through Semi-supervised Topic Modeling. Proceedings of the 17th International World Wide Web Conference (WWW).
- Ma, T. and Wan, X. 2010. Opinion target extraction in Chinese news comments. Proceedings of Coling 2010 Poster Volume (COLING).
- Mei, Q., Ling, X., Wondra, M., Su, H. and Zhai, C. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. Proceedings of International Conference on World Wide Web (WWW).
- Moghaddam, S. and Ester, M. 2011. ILDA: interdependent LDA model for learning latent aspects and their ratings from online product reviews. Proceedings of the Annual ACM SIGIR International conference on Research and Development in Information Retrieval (SIGIR).
- Pang, B. and Lee, L. 2008. Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval.
- Popescu, A. and Etzioni, O. 2005. Extracting product features and opinions from reviews. Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP).
- Qiu, G., Liu, B., Bu, J. and Chen, C. 2011. Opinion Word Expansion and Target Extraction through Double Propagation. Computational Linguistics.
- Ramage, D., Hall, D., Nallapati, R. and Manning, C. 2009. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP).
- Sauper, C., Haghighi, A. and Barzilay, R. 2011. Content models with attitude. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL).
- Somasundaran, S. and Wiebe, J. 2009. Recognizing stances in online debates, Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP.
- Teh, Y., Jordan, M., Beal, M. and Blei, D. 2006. Hierarchical Dirichlet Processes. In Journal of the American Statistical Association (JASA).
- Titov, I. and McDonald, R. 2008. Modeling online reviews with multi-grain topic models. Proceedings of International Conference on World Wide Web (WWW).
- Wallach, H., Mimno, D. and McCallum, A. 2009. Rethinking LDA: Why priors matter. In Neural Information Processing Systems (NIPS).
- Wang, H., Lu, Y. and Zhai, C. 2010. Latent aspect rating analysis on review text data: a rating regression approach. Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).
- Wu, Y., Zhang, Q., Huang, X. and Wu, L. 2009. Phrase dependency parsing for opinion mining. Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP).
- Yi, J., Nasukawa, T., Bunescu, R. and Niblack, W. 2003. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. Proceedings of IEEE International Conference on Data Mining (ICDM).
- Yu, J., Zha, Z. J., Wang, M. and Chua, T. S. 2011. Aspect ranking: identifying important product aspects from online consumer reviews. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics (ACL).
- Zhai, Z., Liu, B. Xu, H. and Jia, P. 2010. Grouping Product Features Using Semi-Supervised Learning with Soft-Constraints. Proceedings of International Conference on Computational Linguistics (COLING).
- Zhai, Z., Liu, B. Xu, H. and Jia, P. 2011. Constrained LDA for Grouping Product Features in Opinion Mining. Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD).
- Zhao, W., Jiang, J., Yan, Y. and Li, X. 2010. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP).
- Zhuang, L., Jing, F. and Zhu, X. 2006. Movie review mining and summarization. Proceedings of International Conference on Information and Knowledge Management (CIKM).



# Learning to “Read Between the Lines” using Bayesian Logic Programs

Sindhu Raghavan Raymond J. Mooney Hyeonseo Ku

Department of Computer Science

The University of Texas at Austin

1616 Guadalupe, Suite 2.408

Austin, TX 78701, USA

{sindhu, mooney, yorq}@cs.utexas.edu

## Abstract

Most information extraction (IE) systems identify facts that are explicitly stated in text. However, in natural language, some facts are implicit, and identifying them requires “reading between the lines”. Human readers naturally use common sense knowledge to *infer* such implicit information from the explicitly stated facts. We propose an approach that uses Bayesian Logic Programs (BLPs), a statistical relational model combining first-order logic and Bayesian networks, to infer additional implicit information from extracted facts. It involves learning uncertain commonsense knowledge (in the form of probabilistic first-order rules) from natural language text by mining a large corpus of automatically extracted facts. These rules are then used to derive additional facts from extracted information using BLP inference. Experimental evaluation on a benchmark data set for machine reading demonstrates the efficacy of our approach.

## 1 Introduction

The task of information extraction (IE) involves automatic extraction of typed entities and relations from unstructured text. IE systems (Cowie and Lehnert, 1996; Sarawagi, 2008) are trained to extract facts that are stated explicitly in text. However, some facts are implicit, and human readers naturally “read between the lines” and *infer* them from the stated facts using commonsense knowledge. Answering many queries can require inferring such implicitly stated facts. Consider the text “Barack Obama is the

president of the United States of America.” Given the query “Barack Obama is a citizen of what country?”, standard IE systems cannot identify the answer since citizenship is not explicitly stated in the text. However, a human reader possesses the commonsense knowledge that the president of a country is almost always a citizen of that country, and easily infers the correct answer.

The standard approach to inferring implicit information involves using commonsense knowledge in the form of logical rules to deduce additional information from the extracted facts. Since manually developing such a knowledge base is difficult and arduous, an effective alternative is to automatically *learn* such rules by mining a substantial database of facts that an IE system has already automatically extracted from a large corpus of text (Nahm and Mooney, 2000). Most existing rule learners assume that the training data is largely accurate and complete. However, the facts extracted by an IE system are always quite noisy and incomplete. Consequently, a purely logical approach to learning and inference is unlikely to be effective. Consequently, we propose using *statistical relational learning* (SRL) (Getoor and Taskar, 2007), specifically, Bayesian Logic Programs (BLPs) (Kersting and De Raedt, 2007), to learn probabilistic rules in first-order logic from a large corpus of extracted facts and then use the resulting BLP to make effective probabilistic inferences when interpreting new documents.

We have implemented this approach by using an off-the-shelf IE system and developing novel adaptations of existing learning methods to efficiently construct fast and effective BLPs for “reading be-

tween the lines.” We present an experimental evaluation of our resulting system on a realistic test corpus from DARPA’s Machine Reading project, and demonstrate improved performance compared to a purely logical approach based on Inductive Logic Programming (ILP) (Lavrač and Džeroski, 1994), and an alternative SRL approach based on Markov Logic Networks (MLNs) (Domingos and Lowd, 2009).

To the best of our knowledge, this is the first paper that employs BLPs for inferring implicit information from natural language text. We demonstrate that it is possible to learn the structure and the parameters of BLPs automatically using only noisy extractions from natural language text, which we then use to infer additional facts from text.

The rest of the paper is organized as follows. Section 2 discusses related work and highlights key differences between our approach and existing work. Section 3 provides a brief background on BLPs. Section 4 describes our BLP-based approach to learning to infer implicit facts. Section 5 describes our experimental methodology and discusses the results of our evaluation. Finally, Section 6 discusses potential future work and Section 7 presents our final conclusions.

## 2 Related Work

Several previous projects (Nahm and Mooney, 2000; Carlson et al., 2010; Schoenmackers et al., 2010; Doppa et al., 2010; Sorower et al., 2011) have mined inference rules from data automatically extracted from text by an IE system. Similar to our approach, these systems use the learned rules to infer additional information from facts directly extracted from a document. Nahm and Mooney (2000) learn *propositional rules* using C4.5 (Quinlan, 1993) from data extracted from computer-related job-postings, and therefore cannot learn multi-relational rules with quantified variables. Other systems (Carlson et al., 2010; Schoenmackers et al., 2010; Doppa et al., 2010; Sorower et al., 2011) learn *first-order rules* (i.e. Horn clauses in first-order logic).

Carlson et al. (2010) modify an ILP system similar to FOIL (Quinlan, 1990) to learn rules with probabilistic conclusions. They use purely logical deduction (forward-chaining) to infer additional facts.

Unlike BLPs, this approach does not use a well-founded probabilistic graphical model to compute coherent probabilities for inferred facts. Further, Carlson et al. (2010) used a human judge to manually evaluate the quality of the learned rules before using them to infer additional facts. Our approach, on the other hand, is completely automated and learns fully parameterized rules in a well-defined probabilistic logic.

Schoenmackers et al. (2010) develop a system called SHERLOCK that uses statistical relevance to learn first-order rules. Unlike our system and others (Carlson et al., 2010; Doppa et al., 2010; Sorower et al., 2011) that use a pre-defined ontology, they automatically identify a set of entity types and relations using “open IE.” They use HOLMES (Schoenmackers et al., 2008), an inference engine based on MLNs (Domingos and Lowd, 2009) (an SRL approach that combines first-order logic and Markov networks) to infer additional facts. However, MLNs include all possible type-consistent groundings of the rules in the corresponding Markov net, which, for larger datasets, can result in an intractably large graphical model. To overcome this problem, HOLMES uses a specialized model construction process to control the grounding process. Unlike MLNs, BLPs naturally employ a more “focused” approach to grounding by including only those literals that are directly relevant to the query.

Doppa et al. (2010) use FARMER (Nijssen and Kok, 2003), an existing ILP system, to learn first-order rules. They propose several approaches to score the rules, which are used to infer additional facts using purely logical deduction. Sorower et al. (2011) propose a probabilistic approach to modeling implicit information as missing facts and use MLNs to infer these missing facts. They learn first-order rules for the MLN by performing exhaustive search. As mentioned earlier, inference using both these approaches, logical deduction and MLNs, have certain limitations, which BLPs help overcome.

DIRT (Lin and Pantel, 2001) and RESOLVER (Yates and Etzioni, 2007) learn inference rules, also called entailment rules that capture synonymous relations and entities from text. Berant et al. (Berant et al., 2011) propose an approach that uses transitivity constraints for learning entailment rules for typed predicates. Unlike the systems described above,

these systems do not learn complex first-order rules that capture common sense knowledge. Further, most of these systems do not use extractions from an IE system to learn entailment rules, thereby making them less related to our approach.

### 3 Bayesian Logic Programs

Bayesian logic programs (BLPs) (Kersting and De Raedt, 2007; Kersting and Raedt, 2008) can be considered as templates for constructing *directed* graphical models (Bayes nets). Formally, a BLP consists of a set of *Bayesian clauses*, definite clauses of the form  $a|a_1, a_2, a_3, \dots, a_n$ , where  $n \geq 0$  and  $a, a_1, a_2, a_3, \dots, a_n$  are *Bayesian predicates* (defined below), and where  $a$  is called the head of the clause ( $\text{head}(c)$ ) and  $(a_1, a_2, a_3, \dots, a_n)$  is the body ( $\text{body}(c)$ ). When  $n = 0$ , a Bayesian clause is a fact. Each Bayesian clause  $c$  is assumed to be universally quantified and range restricted, i.e.  $\text{variables}\{\text{head}\} \subseteq \text{variables}\{\text{body}\}$ , and has an associated *conditional probability table*  $\text{CPT}(c) = P(\text{head}(c)|\text{body}(c))$ . A *Bayesian predicate* is a predicate with a finite domain, and each ground atom for a Bayesian predicate represents a random variable. Associated with each Bayesian predicate is a combining rule such as *noisy-or* or *noisy-and* that maps a finite set of CPTs into a single CPT.

Given a knowledge base as a BLP, standard logical inference (SLD resolution) is used to automatically construct a Bayes net for a given problem. More specifically, given a set of facts and a query, all possible Horn-clause proofs of the query are constructed and used to build a Bayes net for answering the query. The probability of a joint assignment of truth values to the final set of ground propositions is defined as follows:

$$P(X) = \prod_i P(X_i|Pa(X_i)),$$

where  $X = X_1, X_2, \dots, X_n$  represents the set of random variables in the network and  $Pa(X_i)$  represents the parents of  $X_i$ . Once a ground network is constructed, standard probabilistic inference methods can be used to answer various types of queries as reviewed by Koller and Friedman (2009). The parameters in the BLP model can be learned using the methods described by Kersting and De Raedt (2008).

## 4 Learning BLPs to Infer Implicit Facts

### 4.1 Learning Rules from Extracted Data

The first step involves learning commonsense knowledge in the form of first-order Horn rules from text. We first extract facts that are explicitly stated in the text using SIRE (Florian et al., 2004), an IE system developed by IBM. We then learn first-order rules from these extracted facts using LIME (McCreath and Sharma, 1998), an ILP system designed for noisy training data.

We first identify a set of target relations we want to infer. Typically, an ILP system takes a set of positive and negative instances for a target relation, along with a background knowledge base (in our case, other facts extracted from the same document) from which the positive instances are potentially inferable. In our task, we only have direct access to positive instances of target relations, i.e. the relevant facts extracted from the text. So we artificially generate negative instances using the *closed world assumption*, which states that any instance of a relation that is not extracted can be considered a negative instance. While there are exceptions to this assumption, it typically generates a useful (if noisy) set of negative instances. For each relation, we generate all possible type-consistent instances using all constants in the domain. All instances that are not extracted facts (i.e. positive instances) are labeled as negative. The total number of such closed-world negatives can be intractably large, so we randomly sample a fixed-size subset. The ratio of 1:20 for positive to negative instances worked well in our approach.

Since LIME can learn rules using only positive instances, or both positive and negative instances, we learn rules using both settings. We include all unique rules learned from both settings in the final set, since the goal of this step is to learn a large set of potentially useful rules whose relative strengths will be determined in the next step of parameter learning. Other approaches could also be used to learn candidate rules. We initially tried using the popular ALEPH ILP system (Srinivasan, 2001), but it did not produce useful rules, probably due to the high level of noise in our training data.

## 4.2 Learning BLP Parameters

The parameters of a BLP include the CPT entries associated with the Bayesian clauses and the parameters of combining rules associated with the Bayesian predicates. For simplicity, we use a deterministic logical-and model to encode the CPT entries associated with Bayesian clauses, and use *noisy-or* to combine evidence coming from multiple ground rules that have the same head (Pearl, 1988). The noisy-or model requires just a single parameter for each rule, which can be learned from training data.

We learn the noisy-or parameters using the EM algorithm adapted for BLPs by Kersting and De Raedt (2008). In our task, the supervised training data consists of facts that are extracted from the natural language text. However, we usually do not have evidence for inferred facts as well as noisy-or nodes. As a result, there are a number of variables in the ground networks which are always hidden, and hence EM is appropriate for learning the requisite parameters from the partially observed training data.

## 4.3 Inference of Additional Facts using BLPs

Inference in the BLP framework involves backward chaining (Russell and Norvig, 2003) from a specified query (SLD resolution) to obtain all possible deductive proofs for the query. In our context, each target relation becomes a query on which we backchain. We then construct a ground Bayesian network using the resulting deductive proofs for all target relations and learned parameters using the standard approach described in Section 3. Finally, we perform standard probabilistic inference to estimate the marginal probability of each inferred fact. Our system uses Sample Search (Gogate and Dechter, 2007), an approximate sampling algorithm developed for Bayesian networks with deterministic constraints (0 values in CPTs). We tried several exact and approximate inference algorithms on our data, and this was the method that was both tractable and produced the best results.

# 5 Experimental Evaluation

## 5.1 Data

For evaluation, we used DARPA’s machine-reading intelligence-community (IC) data set, which consists of news articles on terrorist events around the

world. There are 10,000 documents each containing an average of 89.5 facts extracted by SIRE (Floridan et al., 2004). SIRE assigns each extracted fact a confidence score and we used only those with a score of 0.5 or higher for learning and inference. An average of 86.8 extractions per document meet this threshold.

DARPA also provides an ontology describing the entities and relations in the IC domain. It consists of 57 entity types and 79 relations. The entity types include Agent, PhysicalThing, Event, TimeLocation, Gender, and Group, each with several subtypes. The type hierarchy is a DAG rather than a tree, and several types have multiple superclasses. For instance, a GeopoliticalEntity can be a HumanAgent as well as a Location. This can cause some problems for systems that rely on a strict typing system, such as MLNs which rely on types to limit the space of ground literals that are considered. Some sample relations are attended-School, approximateNumberOfMembers, mediatingAgent, employs, hasMember, hasMemberHumanAgent, and hasBirthPlace.

## 5.2 Methodology

We evaluated our approach using 10-fold cross validation. We learned first-order rules for the 13 target relations shown in Table 3 from the facts extracted from the training documents (Section 4.1). These relations were selected because the extractor’s recall for them was low. Since LIME does not scale well to large data sets, we could train it on at most about 2,500 documents. Consequently, we split the 9,000 training documents into four disjoint subsets and learned first-order rules from each subset. The final knowledge base included all unique rules learned from any subset. LIME learned several rules that had only entity types in their bodies. Such rules make many incorrect inferences; hence we eliminated them. We also eliminated rules violating type constraints. We learned an average of 48 rules per fold. Table 1 shows some sample learned rules.

We then learned parameters as described in Section 4.2. We initially set all noisy-or parameters to 0.9 based on the intuition that if exactly one rule for a consequent was satisfied, it could be inferred with a probability of 0.9.

$\text{governmentOrganization}(A) \wedge \text{employs}(A,B) \rightarrow \text{hasMember}(A,B)$ <i>If a government organization A employs person B, then B is a member of A</i>
$\text{eventLocation}(A,B) \wedge \text{bombing}(A) \rightarrow \text{thingPhysicallyDamaged}(A,B)$ <i>If a bombing event A took place in location B, then B is physically damaged</i>
$\text{isLedBy}(A,B) \rightarrow \text{hasMemberPerson}(A,B)$ <i>If a group A is led by person B, then B is a member of A</i>
$\text{nationState}(B) \wedge \text{eventLocationGPE}(A,B) \rightarrow \text{eventLocation}(A,B)$ <i>If an event A occurs in a geopolitical entity B, then the event location for that event is B</i>
$\text{mediatingAgent}(A,B) \wedge \text{humanAgentKillingAPerson}(A) \rightarrow \text{killingHumanAgent}(A,B)$ <i>If A is an event in which a human agent is killing a person and the mediating agent of A is an agent B, then B is the human agent that is killing in event A</i>

Table 1: A sample set of rules learned using LIME

For each test document, we performed BLP inference as described in Section 4.3. We ranked all inferences by their marginal probability, and evaluated the results by either choosing the top  $n$  inferences or accepting inferences whose marginal probability was equal to or exceeded a specified threshold. We evaluated two BLPs with different parameter settings: *BLP-Learned-Weights* used noisy-or parameters learned using EM, *BLP-Manual-Weights* used fixed noisy-or weights of 0.9.

### 5.3 Evaluation Metrics

The lack of ground truth annotation for inferred facts prevents an automated evaluation, so we resorted to a manual evaluation. We randomly sampled 40 documents (4 from each test fold), judged the accuracy of the inferences for those documents, and computed *precision*, the fraction of inferences that were deemed correct. For probabilistic methods like BLPs and MLNs that provide certainties for their inferences, we also computed *precision at top n*, which measures the precision of the  $n$  inferences with the highest marginal probability across the 40 test documents. Measuring recall for making inferences is very difficult since it would require labeling a reasonable-sized corpus of documents with *all* of the correct inferences for a given set of target relations, which would be extremely time consuming. Our evaluation is similar to that used in previous related work (Carlson et al., 2010; Schoenmackers et al., 2010).

SIRE frequently makes incorrect extractions, and therefore inferences made from these extractions are also inaccurate. To account for the mistakes made

by the extractor, we report two different precision scores. The “unadjusted” (UA) score, does not correct for errors made by the extractor. The “adjusted” (AD) score does not count mistakes due to extraction errors. That is, if an inference is incorrect because it was based on incorrect extracted facts, we remove it from the set of inferences and calculate precision for the remaining inferences.

### 5.4 Baselines

Since none of the existing approaches have been evaluated on the IC data, we cannot directly compare our performance to theirs. Therefore, we compared to the following methods:

- *Logical Deduction*: This method forward chains on the extracted facts using the first-order rules learned by LIME to infer additional facts. This approach is unable to provide any confidence or probability for its conclusions.
- *Markov Logic Networks (MLNs)*: We use the rules learned by LIME to define the structure of an MLN. In the first setting, which we call *MLN-Learned-Weights*, we learn the MLN’s parameters using the generative weight learning algorithm (Domingos and Lowd, 2009), which we modified to process training examples in an online manner. In online generative learning, gradients are calculated and weights are estimated after processing each example and the learned weights are used as the starting weights for the next example. The pseudo-likelihood of one round is obtained by multiplying the pseudo-likelihood of all examples.

	UA	AD
Precision	29.73 (443/1490)	35.24 (443/1257)

Table 2: Precision for logical deduction. “UA” and “AD” refer to the unadjusted and adjusted scores respectively

In our approach, the initial weights of clauses are set to 10. The average number of iterations needed to acquire the optimal weights is 131. In the second setting, which we call *MLN-Manual-Weights*, we assign a weight of 10 to all rules and maximum likelihood prior to all predicates. *MLN-Manual-Weights* is similar to *BLP-Manual-Weights* in that all rules are given the same weight. We then use the learned rules and parameters to probabilistically infer additional facts using the MC-SAT algorithm implemented in Alchemy,<sup>1</sup> an open-source MLN package.

## 6 Results and Discussion

### 6.1 Comparison to Baselines

Table 2 gives the unadjusted (UA) and adjusted (AD) precision for logical deduction. Out of 1,490 inferences for the 40 evaluation documents, 443 were judged correct, giving an unadjusted precision of 29.73%. Out of these 1,490 inferences, 233 were determined to be incorrect due to extraction errors, improving the adjusted precision to a modest 35.24%.

MLNs made about 127,000 inferences for the 40 evaluation documents. Since it is not feasible to manually evaluate *all* the inferences made by the MLN, we calculated precision using only the top 1000 inferences. Figure 1 shows both unadjusted and adjusted precision at top- $n$  for various values of  $n$  for different BLP and MLN models. For both BLPs and MLNs, simple manual weights result in superior performance than the learned weights. Despite the fairly large size of the overall training sets (9,000 documents), the amount of data for each target relation is apparently still not sufficient to learn particularly accurate weights for both BLPs and MLNs. However, for BLPs, learned weights do show a substantial improvement initially (i.e.

top 25–50 inferences), with an average of 1 inference per document at 91% adjusted precision as opposed to an average of 5 inferences per document at 85% adjusted precision for *BLP-Manual-Weights*. For MLNs, learned weights show a small improvement initially only with respect to adjusted precision. Between BLPs and MLNs, BLPs perform substantially better than MLNs at most points in the curve. However, *MLN-Manual-Weights* improve marginally over *BLP-Learned-Weights* at later points (top 600 and above) on the curve, where the precision is generally very low. Here, the superior performance of BLPs over MLNs could be possibly due to the focused grounding used in the BLP framework.

For BLPs, as  $n$  increases towards including all of the logically sanctioned inferences, as expected, the precision converges to the results for logical deduction. However, as  $n$  decreases, both adjusted and unadjusted precision increase fairly steadily. This demonstrates that probabilistic BLP inference provides a clear improvement over logical deduction, allowing the system to accurately select the best inferences that are most likely to be correct. Unlike the two BLP models, *MLN-Manual-Weights* has more or less the same performance at most points on the curve, and it is slightly better than that of purely-logical deduction. *MLN-Learned-Weights* is worse than purely-logical deduction at most points on the curve.

### 6.2 Results for Individual Target Relations

Table 3 shows the *adjusted* precision for each relation for instances inferred using logical deduction, *BLP-Manual-Weights* and *BLP-Learned-Weights* with a confidence threshold of 0.95. The probabilities estimated for inferences by MLNs are not directly comparable to those estimated by BLPs. As a result, we do not include results for MLNs here. For this evaluation, using a confidence threshold based cutoff is more appropriate than using top- $n$  inferences made by the BLP models since the estimated probabilities can be directly compared across target relations.

For logical deduction, precision is high for a few relations like *employs*, *hasMember*, and *hasMemberHumanAgent*, indicating that the rules learned for these relations are more accurate than the ones

<sup>1</sup><http://alchemy.cs.washington.edu/>

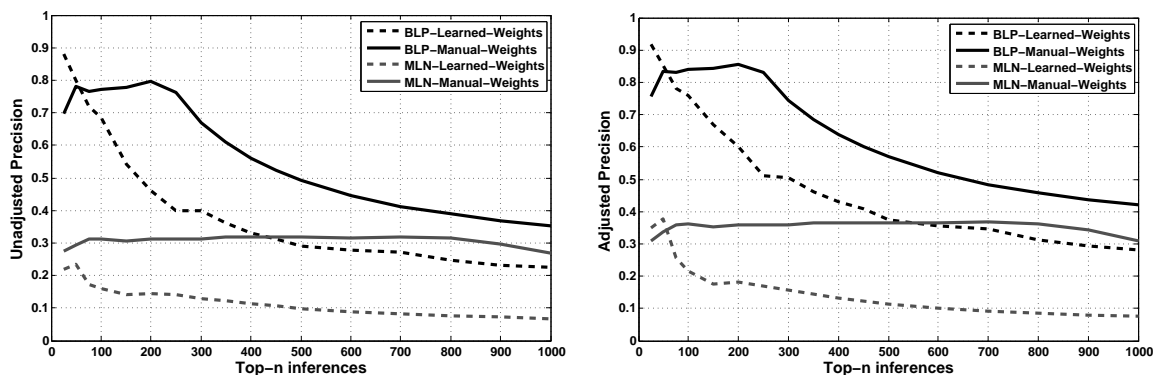


Figure 1: Unadjusted and adjusted precision at top- $n$  for different BLP and MLN models for various values of  $n$

learned for the remaining relations. Unlike relations like `hasMember` that are easily inferred from relations like `employs` and `isLedBy`, certain relations like `hasBirthPlace` are not easily inferable using the information in the ontology. As a result, it might not be possible to learn accurate rules for such target relations. Other reasons include the lack of a sufficiently large number of target-relation instances during training and lack of strictly defined types in the IC ontology.

Both BLP-Manual-Weights and BLP-Learned-Weights also have high precision for several relations (`eventLocation`, `hasMemberHumanAgent`, `thingPhysicallyDamaged`). However, the actual number of inferences can be fairly low. For instance, 103 instances of `hasMemberHumanAgent` are inferred by logical deduction (i.e. 0 confidence threshold), but only 2 of them are inferred by BLP-Learned-Weights at 0.95 confidence threshold, indicating that the parameters learned for the corresponding rules are not very high. For several relations like `hasMember`, `hasMemberPerson`, and `employs`, no instances were inferred by BLP-Learned-Weights at 0.95 confidence threshold. Lack of sufficient training instances (extracted facts) is possibly the reason for learning low weights for such rules. On the other hand, BLP-Manual-Weights has inferred 26 instances of `hasMemberHumanAgent`, out of which all are correct. These results therefore demonstrate the need for sufficient training examples to learn accurate parameters.

### 6.3 Discussion

We now discuss the potential reasons for BLP's superior performance compared to other approaches. Probabilistic reasoning used in BLPs allows for a principled way of determining the most confident inferences, thereby allowing for improved precision over purely logical deduction. The primary difference between BLPs and MLNs lies in the approaches used to construct the ground network. In BLPs, only propositions that can be logically deduced from the extracted evidence are included in the ground network. On the other hand, MLNs include all possible type-consistent groundings of all rules in the network, introducing many ground literals which cannot be logically deduced from the evidence. This generally results in several incorrect inferences, thereby yielding poor performance.

Even though learned weights in BLPs do not result in a superior performance, learned weights in MLNs are substantially worse. Lack of sufficient training data is one of the reasons for learning less accurate weights by the MLN weight learner. However, a more important issue is due to the use of the closed world assumption during learning, which we believe is adversely impacting the weights learned. As mentioned earlier, for the task considered in the paper, if a fact is not explicitly stated in text, and hence not extracted by the extractor, it does not necessarily imply that it is not true. Since existing weight learning approaches for MLNs do not deal with missing data and open world assumption, developing such approaches is a topic for future work.

Apart from developing novel approaches for

Relation	Logical Deduction	BLP-Manual-Weights-.95	BLP-Learned-Weights-.95	No. training instances
employs	<b>69.44</b> (25/36)	<b>92.85</b> (13/14)	nil (0/0)	<b>18440</b>
eventLocation	18.75 (18/96)	<b>100.00</b> (1/1)	<b>100</b> (1/1)	6902
hasMember	<b>95.95</b> (95/99)	<b>97.26</b> (71/73)	nil (0/0)	1462
hasMemberPerson	43.75 (42/96)	<b>100.00</b> (14/14)	nil (0/0)	705
isLedBy	12.30 (8/65)	nil (0/0)	nil (0/0)	8402
mediatingAgent	19.73 (15/76)	nil (0/0)	nil (0/0)	<b>92998</b>
thingPhysicallyDamaged	25.72 (62/241)	<b>90.32</b> (28/31)	<b>90.32</b> (28/31)	<b>24662</b>
hasMemberHumanAgent	<b>95.14</b> (98/103)	<b>100.00</b> (26/26)	<b>100.00</b> (2/2)	3619
killingHumanAgent	15.35 (43/280)	33.33 (2/6)	<b>66.67</b> (2/3)	3341
hasBirthPlace	0 (0/88)	nil (0/0)	nil (0/0)	89
thingPhysicallyDestroyed	nil (0/0)	nil (0/0)	nil (0/0)	800
hasCitizenship	48.05 (37/77)	58.33 (35/60)	nil (0/0)	222
attendedSchool	nil (0/0)	nil (0/0)	nil (0/0)	2

Table 3: Adjusted precision for individual relations (highest values are in bold)

weight learning, additional engineering could potentially improve the performance of MLNs on the IC data set. Due to MLN’s grounding process, several spurious facts like `employs(a,a)` were inferred. These inferences can be prevented by including additional clauses in the MLN that impose integrity constraints that prevent such nonsensical propositions. Further, techniques proposed by Sorower et al. (2011) can be incorporated to explicitly handle missing information in text. Lack of strict typing on the arguments of relations in the IC ontology has also resulted in inferior performance of the MLNs. To overcome this, relations that do not have strictly defined types could be specialized. Finally, we could use the deductive proofs constructed by BLPs to constrain the ground Markov network, similar to the model-construction approach adopted by Singla and Mooney (2011).

However, in contrast to MLNs, BLPs that use first-order rules that are learned by an off-the-shelf ILP system and given simple intuitive hand-coded weights, are able to provide fairly high-precision inferences that augment the output of an IE system and allow it to effectively “read between the lines.”

## 7 Future Work

A primary goal for future research is developing an on-line structure learner for BLPs that can directly learn probabilistic first-order rules from uncertain training data. This will address important limitations of LIME, which cannot accept uncertainty in the extractions used for training, is not specifically

optimized for learning rules for BLPs, and does not scale well to large datasets. Given the relatively poor performance of BLP parameters learned using EM, tests on larger training corpora of extracted facts and the development of improved parameter-learning algorithms are clearly indicated. We also plan to perform a larger-scale evaluation by employing crowdsourcing to evaluate inferred facts for a bigger corpus of test documents. As described above, a number of methods could be used to improve the performance of MLNs on this task. Finally, it would be useful to evaluate our methods on several other diverse domains.

## 8 Conclusions

We have introduced a novel approach using Bayesian Logic Programs to learn to infer implicit information from facts extracted from natural language text. We have demonstrated that it can learn effective rules from a large database of noisy extractions. Our experimental evaluation on the IC data set demonstrates the advantage of BLPs over logical deduction and an approach based on MLNs.

## Acknowledgements

We thank the SIRE team from IBM for providing SIRE extractions on the IC data set. This research was funded by MURI ARO grant W911NF-08-1-0242 and Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program. Experiments were run on the Mastodon Cluster, provided by NSF grant EIA-0303609.



## References

- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 610–619.
- A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr., and T.M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, pages 1306–1313. AAAI Press.
- Jim Cowie and Wendy Lehnert. 1996. Information extraction. *CACM*, 39(1):80–91.
- P. Domingos and D. Lowd. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool, San Rafael, CA.
- Janardhan Rao Doppa, Mohammad NasrEsfahani, Mohammad S. Sorower, Thomas G. Dietterich, Xiaoli Fern, and Prasad Tadepalli. 2010. Towards learning rules from natural texts. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading (FAM-LbR 2010)*, pages 70–77, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Radu Florian, Hany Hassan, Abraham Ittycheriah, Hongyan Jing, Nanda Kambhatla, Xiaoqiang Luo, Nicolas Nicolov, and Salim Roukos. 2004. A statistical model for multilingual entity detection and tracking. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2004)*, pages 1–8.
- L. Getoor and B. Taskar, editors. 2007. *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA.
- Vibhav Gogate and Rina Dechter. 2007. Samplesearch: A scheme that searches for consistent samples. In *Proceedings of Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*.
- K. Kersting and L. De Raedt. 2007. Bayesian Logic Programming: Theory and tool. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, Cambridge, MA.
- Kristian Kersting and Luc De Raedt. 2008. *Basic principles of learning Bayesian Logic Programs*. Springer-Verlag, Berlin, Heidelberg.
- D. Koller and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.
- Nada Lavrač and Saso Džeroski. 1994. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood.
- Deaking Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4):343–360.
- Eric McCreath and Arun Sharma. 1998. Lime: A system for learning relations. In *Ninth International Workshop on Algorithmic Learning Theory*, pages 336–374. Springer-Verlag.
- Un Yong Nahm and Raymond J. Mooney. 2000. A mutually beneficial integration of data mining and information extraction. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI 2000)*, pages 627–632, Austin, TX, July.
- Siegfried Nijssen and Joost N. Kok. 2003. Efficient frequent query discovery in FARMER. In *Proceedings of the Seventh Conference in Principles and Practices of Knowledge Discovery in Database (PKDD 2003)*, pages 350–362. Springer.
- Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.
- J. Ross Quinlan. 1990. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266.
- J. R. Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.
- Stuart Russell and Peter Norvig. 2003. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Upper Saddle River, NJ, 2 edition.
- S. Sarawagi. 2008. Information extraction. *Foundations and Trends in Databases*, 1(3):261–377.
- Stefan Schoenmackers, Oren Etzioni, and Daniel S. Weld. 2008. Scaling textual inference to the web. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 79–88, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. 2010. Learning first-order Horn clauses from web text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1088–1098, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Parag Singla and Raymond Mooney. 2011. Abductive Markov Logic for plan recognition. In *Twenty-fifth National Conference on Artificial Intelligence*.
- Mohammad S. Sorower, Thomas G. Dietterich, Janardhan Rao Doppa, Orr Walker, Prasad Tadepalli, and Xiaoli Fern. 2011. Inverting Grice’s maxims to learn rules from natural language extractions. In *Proceedings of Advances in Neural Information Processing Systems 24*.
- A. Srinivasan, 2001. *The Aleph manual*. <http://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/>.
- Alexander Yates and Oren Etzioni. 2007. Unsupervised resolution of objects and relations on the web. In *Pro-*

*ceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2007).*

# Collective Generation of Natural Image Descriptions

Polina Kuznetsova, Vicente Ordonez, Alexander C. Berg,  
Tamara L. Berg and Yejin Choi

Department of Computer Science

Stony Brook University

Stony Brook, NY 11794-4400

{pkuznetsova,vordonezroma,aberg,tlberg,ychoi}@cs.stonybrook.edu

## Abstract

We present a holistic data-driven approach to image description generation, exploiting the vast amount of (noisy) parallel image data and associated natural language descriptions available on the web. More specifically, given a query image, we retrieve existing human-composed phrases used to describe visually similar images, then selectively combine those phrases to generate a novel description for the query image. We cast the generation process as constraint optimization problems, collectively incorporating multiple interconnected aspects of language composition for *content planning*, *surface realization* and *discourse structure*. Evaluation by human annotators indicates that our final system generates more semantically correct and linguistically appealing descriptions than two nontrivial baselines.

## 1 Introduction

Automatically describing images in natural language is an intriguing, but complex AI task, requiring accurate computational visual recognition, comprehensive world knowledge, and natural language generation. Some past research has simplified the general image description goal by assuming that relevant text for an image is provided (e.g., Aker and Gaizauskas (2010), Feng and Lapata (2010)). This allows descriptions to be generated using effective summarization techniques with relatively surface level image understanding. However, such text (e.g., news articles

or encyclopedic text) is often only loosely related to an image's specific content and many natural images do not come with associated text for summarization.

In contrast, other recent work has focused more on the visual recognition aspect by detecting content elements (e.g., scenes, objects, attributes, actions, etc) and then composing descriptions from scratch (e.g., Yao et al. (2010), Kulkarni et al. (2011), Yang et al. (2011), Li et al. (2011)), or by retrieving existing whole descriptions from visually similar images (e.g., Farhadi et al. (2010), Ordonez et al. (2011)). For the latter approaches, it is unrealistic to expect that there will always exist a *single complete description* for retrieval that is pertinent to a given query image. For the former approaches, visual recognition first generates an intermediate representation of image content using a set of English words, then language generation constructs a full description by adding function words and optionally applying simple re-ordering. Because the generation process sticks relatively closely to the recognized content, the resulting descriptions often lack the kind of coverage, creativity, and complexity typically found in human-written text.

In this paper, we propose a holistic data-driven approach that combines and extends the best aspects of these previous approaches – a) using visual recognition to directly predict individual image content elements, and b) using retrieval from existing human-composed descriptions to generate natural, creative, and inter-

esting captions. We also lift the restriction of retrieving existing *whole descriptions* by gathering *visually relevant phrases* which we combine to produce novel and query-image specific descriptions. By judiciously exploiting the correspondence between image content elements and phrases, it is possible to generate natural language descriptions that are substantially richer in content and more linguistically interesting than previous work.

At a high level, our approach can be motivated by linguistic theories about the connection between reading activities and writing skills, i.e., substantial reading enriches writing skills, (e.g., Hafiz and Tudor (1989), Tsang (1996)). Analogously, our generation algorithm attains a higher level of linguistic sophistication by *reading* large amounts of descriptive text available online. Our approach is also motivated by language grounding by visual worlds (e.g., Roy (2002), Dindo and Zambuto (2010), Monner and Reggia (2011)), as in our approach the meaning of a phrase in a description is implicitly grounded by the relevant content of the image.

Another important thrust of this work is collective *image-level content-planning*, integrating saliency, content relations, and discourse structure based on statistics drawn from a large image-text parallel corpus. This contrasts with previous approaches that generate multiple sentences without considering discourse flow or redundancy (e.g., Li et al. (2011)). For example, for an image showing a flock of birds, generating a large number of sentences stating the relative position of each bird is probably not useful.

Content planning and phrase synthesis can be naturally viewed as constraint optimization problems. We employ Integer Linear Programming (ILP) as an optimization framework that has been used successfully in other generation tasks (e.g., Clarke and Lapata (2006), Martins and Smith (2009), Woodsend and Lapata (2010)). Our ILP formulation encodes a rich set of linguistically motivated constraints and weights that incorporate multiple aspects of the generation process. Empirical results demonstrate that our final system generates linguistically more appealing and semantically more cor-

rect descriptions than two nontrivial baselines.

## 1.1 System Overview

Our system consists of two parts. For a query image, we first retrieve candidate descriptive phrases from a large image-caption database using measures of visual similarity (§2). We then generate a coherent description from these candidates using ILP formulations for content planning (§4) and surface realization (§5).

## 2 Vision & Phrase Retrieval

For a query image, we retrieve relevant candidate natural language phrases by visually comparing the query image to database images from the SBU Captioned Photo Collection (Ordonez et al., 2011) (1 million photographs with associated human-composed descriptions). Visual similarity for several kinds of image content are used to compare the query image to images from the database, including: 1) object detections for 89 common object categories (Felzenszwalb et al., 2010), 2) scene classifications for 26 common scene categories (Xiao et al., 2010), and 3) region based detections for *stuff* categories (e.g. grass, road, sky) (Ordonez et al., 2011). All content types are pre-computed on the million database photos, and caption parsing is performed using the Berkeley PCFG parser (Petrov et al., 2006; Petrov and Klein, 2007).

Given a query image, we identify content elements present using the above classifiers and detectors and then retrieve phrases referring to those content elements from the database. For example, if we detect a horse in a query image, then we retrieve phrases referring to visually similar horses in the database by comparing the color, texture (Leung and Malik, 1999), or shape (Dalal and Triggs, 2005; Lowe, 2004) of the detected horse to detected horses in the database images. We collect four types of phrases for each query image as follows:

[1] **NPs** We retrieve noun phrases for each query object detection (e.g., “the brown cow”) from database captions using visual similarity between object detections computed as an equally weighted linear combination of  $L_2$  dis-

tances on histograms of *color*, *texton* (Leung and Malik, 1999), *HoG* (Dalal and Triggs, 2005) and *SIFT* (Lowe, 2004) features.

[2] **VPs** We retrieve verb phrases for each query object detection (e.g. “boy running”) from database captions using the same measure of visual similarity as for NPs, but restricting the search to only those database instances whose captions contain a verb phrase referring to the object category.

[3] **Region/Stuff PPs** We collect prepositional phrases for each query stuff detection (e.g. “in the sky”, “on the road”) by measuring visual similarity of appearance (color, texton, HoG) and geometric configuration (object-stuff relative location and distance) between query and database detections.

[4] **Scene PPs** We also collect prepositional phrases referring to general image scene context (e.g. “at the market”, “on hot summer days”, “in Sweden”) based on global scene similarity computed using  $L_2$  distance between scene classification score vectors (Xiao et al., 2010) computed on the query and database images.

### 3 Overview of ILP Formulation

For each image, we aim to generate multiple sentences, each sentence corresponding to a single distinct object detected in the given image. Each sentence comprises of the NP for the main object, and a *subset* of the corresponding VP, region/stuff PP, and scene PP retrieved in §2. We consider four different types of operations to generate the final description for each image:

- T1.** Selecting the set of objects to describe (one object per sentence).
- T2.** Re-ordering sentences (i.e., re-ordering objects).
- T3.** Selecting the set of phrases for each sentence.
- T4.** Re-ordering phrases within each sentence.

The ILP formulation of §4 addresses T1 & T2, i.e., content-planning, and the ILP of §5 addresses T3 & T4, i.e., surface realization.<sup>1</sup>

<sup>1</sup>It is possible to create one conjoined ILP formulation to address all four operations T1—T4 at once. For com-

## 4 Image-level Content Planning

First we describe image-level content planning, i.e., *abstract generation*. The goals are to (1) select a subset of the objects based on saliency and semantically compatibility, and (2) order the selected objects based on their content relations.

### 4.1 Variables and Objective Function

The following set of indicator variables encodes the selection of objects and ordering:

$$y_{sk} = \begin{cases} 1, & \text{if object } s \text{ is selected} \\ & \text{for position } k \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where  $k = 1, \dots, S$  encodes the position (order) of the selected objects, and  $s$  indexes one of the objects. In addition, we define a set of variables indicating specific pairs of adjacent objects:

$$y_{skt(k+1)} = \begin{cases} 1, & \text{if } y_{sk} = y_{t(k+1)} = 1 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

The objective function,  $F$ , that we will maximize is a weighted linear combination of these indicator variables and can be optimized using integer linear programming:

$$F = \sum_s F_s \cdot \sum_{k=1}^S y_{sk} - \sum_{st} F_{st} \cdot \sum_{k=1}^{S-1} y_{skt(k+1)} \quad (3)$$

where  $F_s$  quantifies the salience/confidence of the object  $s$ , and  $F_{st}$  quantifies the semantic compatibility between the objects  $s$  and  $t$ . These coefficients (weights) will be described in §4.3 and §4.4. We use IBM CPLEX to optimize this objective function subject to the constraints introduced next in §4.2.

### 4.2 Constraints

**Consistency Constraints:** We enforce consistency between indicator variables for individual objects (Eq. 1) and consecutive objects (Eq. 2) so that  $y_{skt(k+1)} = 1$  iff  $y_{sk} = 1$  and  $y_{t(k+1)} = 1$ :

$$\forall_{stk}, y_{skt(k+1)} \leq y_{sk} \quad (4)$$

$$y_{skt(k+1)} \leq y_{t(k+1)} \quad (5)$$

$$y_{skt(k+1)} + (1 - y_{sk}) + (1 - y_{t(k+1)}) \geq 1 \quad (6)$$

putational and implementation efficiency however, we opt for the two-step approach.

To avoid empty descriptions, we enforce that the result includes at least one object:

$$\sum_s y_{s1} = 1 \quad (7)$$

To enforce contiguous positions be selected:

$$\forall k = 2, \dots, S-1, \quad \sum_s y_{s(k+1)} \leq \sum_s y_{sk} \quad (8)$$

**Discourse constraints:** To avoid spurious descriptions, we allow at most two objects of the same type, where  $c_s$  is the type of object  $s$ :

$$\forall c \in objTypes, \quad \sum_{\{s: c_s=c\}} \sum_{k=1}^S y_{sk} \leq 2 \quad (9)$$

### 4.3 Weight $F_s$ : Object Detection Confidence

In order to quantify the confidence of the object detector for the object  $s$ , we define  $0 \leq F_s \leq 1$  as the mean of the detector scores for that object type in the image.

### 4.4 Weight $F_{st}$ : Ordering and Compatibility

The weight  $0 \leq F_{st} \leq 1$  quantifies the compatibility of the object pairing  $(s, t)$ . Note that in the objective function, we subtract this quantity from the function to be maximized. This way, we create a competing tension between the single object selection scores and the pairwise compatibility scores, so that variable number of objects can be selected.

**Object Ordering Statistics:** People have biases on the order of topic or content flow. We measure these biases by collecting statistics on ordering of object names from the 1 million image descriptions in the SBU Captioned Dataset (Ordonez et al., 2011). Let  $f_{ord}(w_1, w_2)$  be the number of times  $w_1$  appeared before  $w_2$ . For instance,  $f_{ord}(window, house) = 2895$  and  $f_{ord}(house, window) = 1250$ , suggesting that people are more likely to mention a window before mentioning a house/building<sup>2</sup>. We use these ordering statistics to enhance content flow. We define score for the order of objects using Z-score for normalization as follows:

$$\hat{F}_{st} = \frac{f_{ord}(c_s, c_t) - mean(f_{ord})}{std.dev(f_{ord})} \quad (10)$$

<sup>2</sup>We take into account synonyms.

We then transform  $\hat{F}_{st}$  so that  $\hat{F}_{st} \in [0,1]$ , and then set  $F_{st} = 1 - \hat{F}_{st}$  so that smaller values correspond to better choices.

## 5 Surface Realization

Recall that for each image, the computer vision system identifies phrases from descriptions of images that are similar in a variety of aspects. The result is a set of phrases representing four different types of information (§2). From this assortment of phrases, we aim to select a subset and glue them together to compose a complete sentence that is linguistically plausible and semantically truthful to the content of the image.

### 5.1 Variables and Objective Function

The following set of variables encodes the selection of phrases and their ordering in constructing  $S'$  sentences.

$$x_{sijk} = \begin{cases} 1, & \text{if phrase } i \text{ of type } j \\ & \text{is selected} \\ & \text{for position } k \\ & \text{in sentence } s \\ 0, & \text{otherwise} \end{cases} \quad (11)$$

where  $k = 1, \dots, N$  encodes the ordering of the selected phrases, and  $j$  indexes one of the four phrases types (object-NPs, action-VPs, region-PPs, scene-PPs),  $i = 1, \dots, M$  indexes one of the  $M$  candidate phrases of each phrase type, and  $s = 1, \dots, S'$  encodes the sentence (object). In addition, we define indicator variables for adjacent pairs of phrases:  $x_{sijkpq(k+1)} = 1$  if  $x_{sijk} = x_{spq(k+1)} = 1$  and 0 otherwise. Finally, we define the objective function  $F$  as:

$$F = \sum_{sij} F_{sij} \cdot \sum_{k=1}^N x_{sijk} - \sum_{sijpq} F_{sijpq} \cdot \sum_{k=1}^{N-1} x_{sijkpq(k+1)} \quad (12)$$

where  $F_{sij}$  weights individual phrase goodness and  $F_{sijpq}$  adjacent phrase goodness. All coefficients (weights) will be described in Section 5.3 and 5.4.

We optionally prepend the first sentence in a generated description with a *cognitive phrase*.<sup>3</sup>

<sup>3</sup>We collect most frequent 200 phrases of length 1-7 that start a caption from the SBU Captioned Photo Collection.

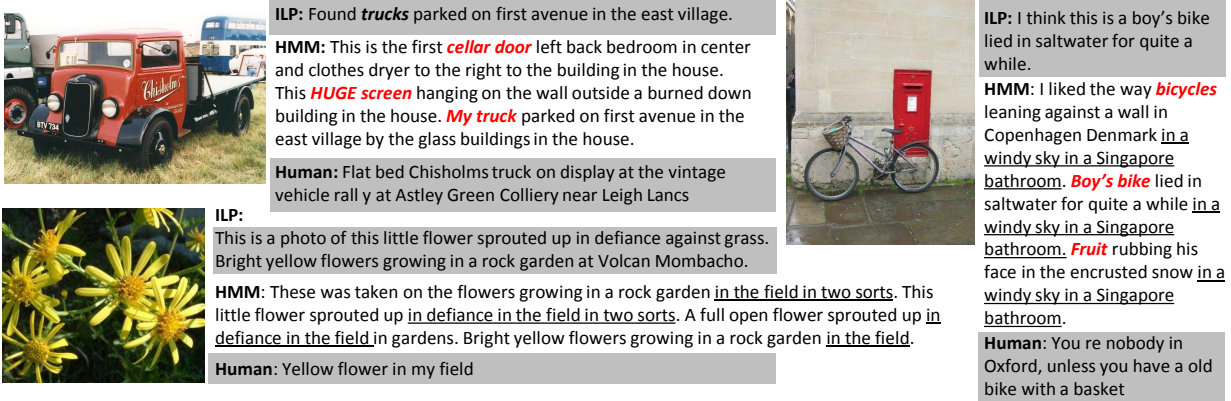


Figure 1: ILP & HMM generated captions. In HMM generated captions, underlined phrases show redundancy across different objects (due to lack of discourse constraints), and phrases in **boldface** show awkward topic flow (due to lack of content planning). Note that in the bicycle image, the visual recognizer detected two separate bicycles and some fruits, as can be seen in the HMM result. Via collective image-level content planning (see §4), some of these erroneous detection can be corrected, as shown in the ILP result. Spurious and redundant phrases can be suppressed via discourse constraints (see §5).

These are generic constructs that are often used to start a description about an image, for instance, “This is an image of...”. We treat these phrases as an additional type, but omit corresponding variables and constraints for brevity.

## 5.2 Constraints

**Consistency Constraints:** First we enforce consistency between the unary variables (Eq. 11) and the pairwise variables so that  $x_{sijkpqm} = 1$  iff  $x_{sijk} = 1$  and  $x_{spqm} = 1$ :

$$\forall ijkpqm, x_{sijkpqm} \leq x_{sijk} \quad (13)$$

$$x_{sijkpqm} \leq x_{spqm} \quad (14)$$

$$x_{sijkpqm} + (1 - x_{sijk}) + (1 - x_{spqm}) \geq 1 \quad (15)$$

Next we include constraints similar to Eq. 8 (contiguous slots are filled), but omit them for brevity. Finally, we add constraints to ensure at least two phrases are selected for each sentence, to promote informative descriptions.

**Linguistic constraints:** We include linguistically motivated constraints to generate syntactically and semantically plausible sentences. First we enforce a noun-phrase to be selected to ensure semantic relevance to the image:

$$\forall s, \sum_{ik} x_{siNpk} = 1 \quad (16)$$

Also, to avoid content redundancy, we allow at most one phrase of each type:

$$\forall sj, \sum_i \sum_{k=1}^N x_{sijk} \leq 1 \quad (17)$$

**Discourse constraints:** We allow at most one prepositional scene phrase for the whole description to avoid redundancy:

$$\text{For } j = PPscene, \sum_{sik} x_{sijk} \leq 1 \quad (18)$$

We add constraints that prevent the inclusion of more than one phrase with identical head words:  $\forall s, ij, pq$  with the same heads,

$$\sum_{k=1}^N x_{sijk} + \sum_{k=1}^N x_{spqk} \leq 1 \quad (19)$$

## 5.3 Unary Phrase Selection

Let  $M_{sij}$  be the confidence score for phrase  $x_{sij}$  given by the image–phrase matching algorithm (§2). To make the scores across different phrase types comparable, we normalize them using Z-score:  $F_{sij} = \text{norm}'(M_{sij}) = (M_{sij} - \text{mean}_j) / \text{dev}_j$ , and then transform the values into the range of [0,1].

## 5.4 Pairwise Phrase Cohesion

In this section, we describe the pairwise phrase cohesion score  $F_{sijpq}$  defined for each  $x_{sijpq}$  in

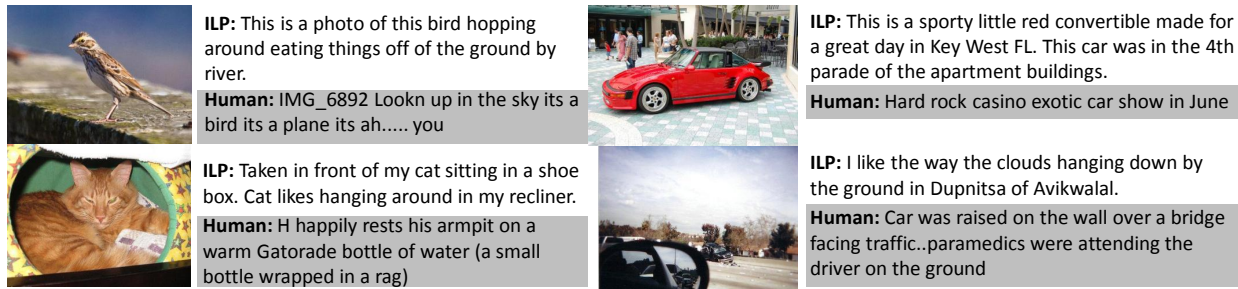


Figure 2: In some cases (16%), ILP generated captions were preferred over human written ones!

the objective function (Eq. 12). Via  $F_{sijpq}$ , we aim to quantify the degree of syntactic and semantic cohesion across two phrases  $x_{sij}$  and  $x_{spq}$ . Note that we subtract this cohesion score from the objective function. This trick helps the ILP solver to generate sentences with varying number of phrases, rather than always selecting the maximum number of phrases allowed.

**N-gram Cohesion Score:** We use n-gram statistics from the Google Web 1-T dataset (Brants and Franz., 2006) Let  $L_{sijpq}$  be the set of all n-grams ( $2 \leq n \leq 5$ ) across  $x_{sij}$  and  $x_{spq}$ . Then the n-gram cohesion score is computed as:

$$F_{sijpq}^{\text{NGRAM}} = 1 - \frac{\sum_{l \in L_{sijpq}} \text{NPMI}(l)}{\text{size}(L_{sijpq})} \quad (20)$$

$$\text{NPMI}(ngr) = \frac{\text{PMI}(ngr) - \text{PMI}_{\min}}{\text{PMI}_{\max} - \text{PMI}_{\min}} \quad (21)$$

Where NPMI is the normalized point-wise mutual information.<sup>4</sup>

**Co-occurrence Cohesion Score:** To capture long-distance cohesion, we introduce a co-occurrence-based score, which measures order-preserved co-occurrence statistics between the head words  $h_{sij}$  and  $h_{spq}$ <sup>5</sup>. Let  $f_{\Sigma}(h_{sij}, h_{spq})$  be the sum frequency of all n-grams that start with  $h_{sij}$ , end with  $h_{spq}$  and contain a preposition  $prep(spq)$  of the phrase  $spq$ . Then the

<sup>4</sup>We include the n-gram cohesion for the sentence boundaries as well, by approximating statistics for sentence boundaries with punctuation marks in the Google Web 1-T data.

<sup>5</sup>For simplicity, we use the last word of a phrase as the head word, except VPs where we take the main verb.

co-occurrence cohesion is computed as:

$$F_{sijpq}^{\text{CO}} = \frac{\max(f_{\Sigma}) - f_{\Sigma}(h_{sij}, h_{spq})}{\max(f_{\Sigma}) - \min(f_{\Sigma})} \quad (22)$$

**Final Cohesion Score:** Finally, the pairwise phrase cohesion score  $F_{ijpq}$  is a weighted sum of n-gram and co-occurrence cohesion scores:

$$F_{sijpq} = \frac{\alpha \cdot F_{sijpq}^{\text{NGRAM}} + \beta \cdot F_{sijpq}^{\text{CO}}}{\alpha + \beta} \quad (23)$$

where  $\alpha$  and  $\beta$  can be tuned via grid search, and  $F_{ijpq}^{\text{NGRAM}}$  and  $F_{ijpq}^{\text{CO}}$  are normalized  $\in [0, 1]$  for comparability. Notice that  $F_{sijpq}$  is in the range  $[0, 1]$  as well.

## 6 Evaluation

**TestSet:** Because computer vision is a challenging and unsolved problem, we restrict our query set to images where we have high confidence that visual recognition algorithms perform well. We collect 1000 test images by running a large number (89) of object detectors on 20,000 images and selecting images that receive confident object detection scores, with some preference for images with multiple object detections to obtain good examples for testing discourse constraints.

**Baselines:** We compare our ILP approaches with two nontrivial baselines: the first is an HMM approach (comparable to Yang et al. (2011)), which takes as input the same set of candidate phrases described in §2, but for decoding, we fix the ordering of phrases as [ NP – VP – Region PP – Scene PP] and find the best combination of phrases using the Viterbi algorithm. We use the same rich set of pairwise



cognitive phrases:	HMM with	HMM w/o	ILP with	ILP w/o
	0.111	0.114	0.114	0.116

Table 1: Automatic Evaluation

	ILP selection rate
ILP V.S. HMM (w/o cogn)	67.2%
ILP V.S. HMM (with cogn)	66.3%

Table 2: Human Evaluation (without images)

	ILP selection rate
ILP V.S. HMM (w/o cogn)	53.17%
ILP V.S. HMM (with cogn)	54.5%
ILP V.S. RETRIEVAL	71.8%
ILP V.S. Human	16%

Table 3: Human Evaluation (with images)

phrase cohesion scores (§5.4) used for the ILP formulation, producing a strong baseline<sup>6</sup>.

The second baseline is a recent RETRIEVAL based description method (Ordonez et al., 2011), that searches the large parallel corpus of images and captions, and transfers a caption from a visually similar database image to the query. This again is a very strong baseline, as it exploits the vast amount of image-caption data, and produces a description high in linguistic quality (since the captions were written by human annotators).

**Automatic Evaluation:** Automatically quantifying the quality of machine generated sentences is known to be difficult. BLEU score (Papineni et al., 2002), despite its simplicity and limitations, has been one of the common choices for automatic evaluation of image descriptions (Farhadi et al., 2010; Kulkarni et al., 2011; Li et al., 2011; Ordonez et al., 2011), as it correlates reasonably well with human evaluation (Belz and Reiter, 2006).

Table 1 shows the the BLEU @1 against the original caption of 1000 images. We see that the ILP improves the score over HMM consistently, with or without the use of cognitive phrases.

<sup>6</sup>Including other long-distance scores in HMM decoding would make the problem NP-hard and require more sophisticated decoding, e.g. ILP.

	Grammar	Cognitive	Relevance
HMM	3.40( $\sigma=.82$ )	3.40( $\sigma=.88$ )	2.25( $\sigma=1.37$ )
ILP	3.56( $\sigma=.90$ )	3.60( $\sigma=.98$ )	2.37( $\sigma=1.49$ )
Hum.	4.36( $\sigma=.79$ )	4.77( $\sigma=.66$ )	3.86( $\sigma=1.60$ )

Table 4: Human Evaluation: Multi-Aspect Rating ( $\sigma$  is a standard deviation)

**Human Evaluation I – Ranking:** We complement the automatic evaluation with Mechanical Turk evaluation. In ranking evaluation, we ask raters to choose a better caption between two choices<sup>7</sup>. We do this rating with and without showing the images, as summarized in Table 2 & 3. When images are shown, raters evaluate content relevance as well as linguistic quality of the captions. Without images, raters evaluate only linguistic quality.

We found that raters generally prefer ILP generated captions over HMM generated ones, twice as much (67.2% ILP V.S. 32.8% HMM), if images are not presented. However the difference is less pronounced when images are shown. There could be two possible reasons. The first is that when images are shown, the Turkers do not try as hard to tell apart the subtle difference between the two imperfect captions. The second is that the relative content relevance of ILP generated captions is negating the superiority in linguistic quality. We explore this question using multi-aspect rating, described below.

Note that ILP generated captions are exceedingly (71.8 %) preferred over the RETRIEVAL baseline (Ordonez et al., 2011), despite the generated captions tendency to be more prone to grammatical and cognitive errors than retrieved ones. This indicates that the generated captions must have substantially better content relevance to the query image, supporting the direction of this research. Finally, notice that as much as 16% of the time, ILP generated captions are preferred over the original human generated ones (examples in Figure 2).

**Human Evaluation II – Multi-Aspect Rating:** Table 4 presents rating in the 1–5 scale (5: perfect, 4: almost perfect, 3: 70~80% good, 2:

<sup>7</sup>We present two captions in a randomized order.

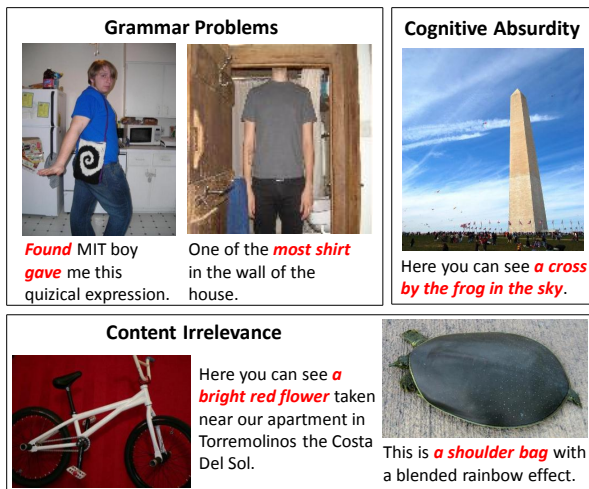


Figure 3: Examples with different aspects of problems in the ILP generated captions.

50~70% good, 1: totally bad) in three different aspects: *grammar*, *cognitive correctness*,<sup>8</sup> and *relevance*. We find that ILP improves over HMM in all aspects, however, the relevance score is noticeably worse than scores of two other criteria. It turns out human raters are generally more critical against the relevance aspect, as can be seen in the ratings given to the original human generated captions.

**Discussion with Examples:** Figure 1 shows contrastive examples of HMM vs ILP generated captions. Notice that HMM captions look *robotic*, containing spurious and redundant phrases due to lack of discourse constraints, and often discussing an awkward set of objects due to lack of image-level content planning. Also notice how image-level content planning underpinned by language statistics helps correct some of the erroneous vision detections. Figure 3 shows some example mistakes in the ILP generated captions.

## 7 Related Work & Discussion

Although not directly focused on image description generation, some previous work in the realm of summarization shares the similar problem of content planning and surface realization. There

<sup>8</sup>E.g., “A desk on top of a cat” is grammatically correct, but cognitively absurd.

are subtle, but important differences however. First, sentence compression is hardly the goal of image description generation, as human written descriptions are not necessarily succinct.<sup>9</sup> Second, unlike summarization, we are not given with a set of coherent text snippet to begin with, and the level of noise coming from the visual recognition errors is much higher than that of starting with clean text. As a result, choosing an additional phrase in the image description is much riskier than it is in summarization.

Some recent research proposed very elegant approaches to summarization using ILP for collective content planning and/or surface realization (e.g., Martins and Smith (2009), Woodsend and Lapata (2010), Woodsend et al. (2010)). Perhaps the most important difference in our approach is the use of *negative* weights in the objective function to create the necessary tension between selection (saliency) and compatibility, which makes it possible for ILP to generate variable length descriptions, effectively correcting some of the erroneous vision detections. In contrast, all previous work operates with a pre-defined upper limit in length, hence the ILP was formulated to include as many textual units as possible modulo constraints.

To conclude, we have presented a collective approach to generating natural image descriptions. Our approach is the first to systematically incorporate state of the art computer vision to retrieve visually relevant candidate phrases, then produce images descriptions that are substantially more complex and human-like than previous attempts.

**Acknowledgments** T. L. Berg is supported in part by NSF CAREER award #1054133; A. C. Berg and Y. Choi are partially supported by the Stony Brook University Office of the Vice President for Research. We thank K. Yamaguchi, X. Han, M. Mitchell, H. Daume III, A. Goyal, K. Stratos, A. Mensch, J. Dodge for data pre-processing and useful initial discussions.

<sup>9</sup>On a related note, the notion of saliency also differs in that human written captions often digress on details that might be tangential to the visible content of the image. E.g., “This is a dress *my mom made*.”, where the picture does not show a woman making the dress.

## References

- Ahmet Aker and Robert Gaizauskas. 2010. Generating image descriptions using dependency relational patterns. In *ACL*.
- Anja Belz and Ehud Reiter. 2006. Comparing automatic and human evaluation of nlg systems. In *EACL 2006, 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, April 3-7, 2006, Trento, Italy*. The Association for Computer Linguistics.
- Thorsten Brants and Alex Franz. 2006. Web 1t 5-gram version 1. In *Linguistic Data Consortium*.
- James Clarke and Mirella Lapata. 2006. Constraint-based sentence compression: An integer programming approach. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 144–151, Sydney, Australia, July. Association for Computational Linguistics.
- Navneet Dalal and Bill Triggs. 2005. Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886–893, Washington, DC, USA. IEEE Computer Society.
- Haris Dindo and Daniele Zambuto. 2010. A probabilistic approach to learning a visually grounded language model through human-robot interaction. In *IROS*, pages 790–796. IEEE.
- Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. 2010. Every picture tells a story: generating sentences for images. In *ECCV*.
- Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. 2010. Object detection with discriminatively trained part based models. *tPAMI*, Sept.
- Yansong Feng and Mirella Lapata. 2010. How many words is a picture worth? automatic caption generation for news images. In *ACL*.
- Fateh Muhammad Hafiz and Ian Tudor. 1989. Extensive reading and the development of language skills. *ELT Journal*, 43(1):4–13.
- Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. 2011. Babytalk: Understanding and generating simple image descriptions. In *CVPR*.
- Thomas K. Leung and Jitendra Malik. 1999. Recognizing surfaces using three-dimensional textons. In *ICCV*.
- Siming Li, Girish Kulkarni, Tamara L. Berg, Alexander C. Berg, and Yejin Choi. 2011. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 220–228, Portland, Oregon, USA, June. Association for Computational Linguistics.
- David G. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, November.
- Andre Martins and Noah A. Smith. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 1–9, Boulder, Colorado, June. Association for Computational Linguistics.
- Derek D. Monner and James A. Reggia. 2011. Systematically grounding language through vision in a deep, recurrent neural network. In *Proceedings of the 4th international conference on Artificial general intelligence, AGI'11*, pages 112–121, Berlin, Heidelberg. Springer-Verlag.
- Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. 2011. Im2text: Describing images using 1 million captioned photographs. In *Neural Information Processing Systems (NIPS)*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *HLT-NAACL*.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *COLING/ACL*.
- Deb K. Roy. 2002. Learning visually-grounded words and syntax for a scene description task. *Computer Speech and Language*, In review.
- Wai-King Tsang. 1996. Comparing the effects of reading and writing on writing performance. *Applied Linguistics*, 17(2):210–233.
- Kristian Woodsend and Mirella Lapata. 2010. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574, Uppsala, Sweden, July. Association for Computational Linguistics.
- Kristian Woodsend, Yansong Feng, and Mirella Lapata. 2010. Title generation with quasi-synchronous grammar. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pages 513–523, Stroudsburg, PA, USA. Association for Computational Linguistics.

- Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. 2010. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*.
- Yezhou Yang, Ching Teo, Hal Daume III, and Yianis Aloimonos. 2011. Corpus-guided sentence generation of natural images. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 444–454, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Benjamin Z. Yao, Xiong Yang, Liang Lin, Mun Wai Lee, and Song-Chun Zhu. 2010. I2t: Image parsing to text description. *Proc. IEEE*, 98(8).

# Concept-to-text Generation via Discriminative Reranking

Ioannis Konstas and Mirella Lapata

Institute for Language, Cognition and Computation

School of Informatics, University of Edinburgh

10 Crichton Street, Edinburgh EH8 9AB

i.konstas@sms.ed.ac.uk, mlap@inf.ed.ac.uk

## Abstract

This paper proposes a data-driven method for concept-to-text generation, the task of automatically producing textual output from non-linguistic input. A key insight in our approach is to reduce the tasks of content selection (“what to say”) and surface realization (“how to say”) into a common parsing problem. We define a probabilistic context-free grammar that describes the structure of the input (a corpus of database records and text describing some of them) and represent it compactly as a weighted hypergraph. The hypergraph structure encodes exponentially many derivations, which we rerank discriminatively using local and global features. We propose a novel decoding algorithm for finding the best scoring derivation and generating in this setting. Experimental evaluation on the ATIS domain shows that our model outperforms a competitive discriminative system both using BLEU and in a judgment elicitation study.

## 1 Introduction

Concept-to-text generation broadly refers to the task of automatically producing textual output from non-linguistic input such as databases of records, logical form, and expert system knowledge bases (Reiter and Dale, 2000). A variety of concept-to-text generation systems have been engineered over the years, with considerable success (e.g., Dale et al. (2003), Reiter et al. (2005), Green (2006), Turner et al. (2009)). Unfortunately, it is often difficult to adapt them across different domains as they rely mostly on handcrafted components.

In this paper we present a data-driven approach to concept-to-text generation that is domain-independent, conceptually simple, and flexible. Our generator learns from a set of database records and textual descriptions (for some of them). An example from the air travel domain is shown in Figure 1. Here, the records provide a structured representation of the flight details (e.g., departure and arrival time, location), and the text renders some of this information in natural language. Given such input, our model determines which records to talk about (*content selection*) and which words to use for describing them (*surface realization*). Rather than breaking up the generation process into a sequence of local decisions, we perform both tasks jointly. A key insight in our approach is to reduce content selection and surface realization into a common parsing problem. Specifically, we define a probabilistic context-free grammar (PCFG) that captures the structure of the database and its correspondence to natural language. This grammar represents multiple derivations which we encode compactly using a weighted hypergraph (or packed forest), a data structure that defines a weight for each tree.

Following a generative approach, we could first learn the weights of the PCFG by maximising the joint likelihood of the model and then perform generation by finding the best derivation tree in the hypergraph. The performance of this baseline system could be potentially further improved using discriminative reranking (Collins, 2000). Typically, this method first creates a list of  $n$ -best candidates from a generative model, and then reranks them with arbitrary features (both local and global) that are either not computable or intractable to compute within the

	<b>Flight</b>	<b>Day Number</b>	<b>Month</b>	<b>Condition</b>	<b>Search</b>
Database:	<u>from to</u> denver boston	<u>number dep/ar</u> 9 departure	<u>month dep/ar</u> august departure	<u>arg1 arg2 type</u> arrival_time 1600 <	<u>type what</u> query flight
$\lambda$ -expression:	$\lambda x. flight(x) \wedge from(x, denver) \wedge to(x, boston) \wedge day\_number(x, 9) \wedge month(x, august) \wedge less\_than(arrival\_time(x), 1600)$				
Text:	Give me the flights leaving Denver August ninth coming back to Boston before 4pm.				

Figure 1: Example of non-linguistic input as a structured database and logical form and its corresponding text. We omit record fields that have no value, for the sake of brevity.

baseline system.

An appealing alternative is to rerank the hypergraph *directly* (Huang, 2008). As it compactly encodes exponentially many derivations, we can explore a much larger hypothesis space than would have been possible with an  $n$ -best list. Importantly, in this framework non-local features are computed at all internal hypergraph nodes, allowing the decoder to take advantage of them continuously at all stages of the generation process. We incorporate features that are local with respect to a span of a sub-derivation in the packed forest; we also (approximately) include features that arbitrarily exceed span boundaries, thus capturing more global knowledge. Experimental results on the ATIS domain (Dahl et al., 1994) demonstrate that our model outperforms a baseline based on the best derivation and a state-of-the-art discriminative system (Angeli et al., 2010) by a wide margin.

Our contributions in this paper are threefold: we recast concept-to-text generation in a probabilistic parsing framework that allows to jointly optimize content selection and surface realization; we represent parse derivations compactly using hypergraphs and illustrate the use of an algorithm for *generating* (rather than parsing) in this framework; finally, the application of discriminative reranking to concept-to-text generation is novel to our knowledge and as our experiments show beneficial.

## 2 Related Work

Early discriminative approaches to text generation were introduced in spoken dialogue systems, and usually tackled content selection and surface realization separately. Ratnaparkhi (2002) conceptualized surface realization (from a fixed meaning representation) as a classification task. Local and non-local information (e.g., word  $n$ -grams, long-

range dependencies) was taken into account with the use of features in a maximum entropy probability model. More recently, Wong and Mooney (2007) describe an approach to surface realization based on synchronous context-free grammars. The latter are learned using a log-linear model with minimum error rate training (Och, 2003).

Angeli et al. (2010) were the first to propose a unified approach to content selection and surface realization. Their model operates over automatically induced alignments of words to database records (Liang et al., 2009) and decomposes into a sequence of discriminative local decisions. They first determine which records in the database to talk about, then which fields of those records to mention, and finally which words to use to describe the chosen fields. Each of these decisions is implemented as a log-linear model with features learned from training data. Their surface realization component performs decisions based on templates that are automatically extracted and smoothed with domain-specific knowledge in order to guarantee fluent output.

Discriminative reranking has been employed in many NLP tasks such as syntactic parsing (Charniak and Johnson, 2005; Huang, 2008), machine translation (Shen et al., 2004; Li and Khudanpur, 2009) and semantic parsing (Ge and Mooney, 2006). Our model is closest to Huang (2008) who also performs forest reranking on a hypergraph, using both local and non-local features, whose weights are tuned with the averaged perceptron algorithm (Collins, 2002). We adapt forest reranking to generation and introduce several task-specific features that boost performance. Although conceptually related to Angeli et al. (2010), our model optimizes content selection and surface realization simultaneously, rather than as a sequence. The discriminative aspect of two models is also fundamentally different. We have a single reranking component that applies

throughout, whereas they train different discriminative models for each local decision.

### 3 Problem Formulation

We assume our generator takes as input a set of database records  $\mathbf{d}$  and produces text  $\mathbf{w}$  that verbalizes some of these records. Each record  $r \in \mathbf{d}$  has a type  $r.t$  and a set of fields  $f$  associated with it. Fields have different values  $f.v$  and types  $f.t$  (i.e., integer or categorical). For example, in Figure 1, *flight* is a record type with fields *from* and *to*. The values of these fields are `denver` and `boston` and their type is categorical.

During training, our algorithm is given a corpus consisting of several *scenarios*, i.e., database records paired with texts like those shown in Figure 1. The database (and accompanying texts) are next converted into a PCFG whose weights are learned from training data. PCFG derivations are represented as a weighted directed hypergraph (Gallo et al., 1993). The weights on the hyperarcs are defined by a variety of feature functions, which we learn via a discriminative online update algorithm. During testing, we are given a set of database records without the corresponding text. Using the learned feature weights, we compile a hypergraph specific to this test input and decode it approximately (Huang, 2008). The hypergraph representation allows us to decompose the feature functions and compute them piecemeal at each hyperarc (or sub-derivation), rather than at the root node as in conventional  $n$ -best list reranking. Note that the algorithm does not separate content selection from surface realization, both subtasks are optimized jointly through the probabilistic parsing formulation.

#### 3.1 Grammar Definition

We capture the structure of the database with a number of CFG rewrite rules, in a similar way to how Liang et al. (2009) define Markov chains in their hierarchical model. These rules are purely syntactic (describing the intuitive relationship between records, records and fields, fields and corresponding words), and could apply to any database with similar structure irrespectively of the semantics of the domain.

Our grammar is defined in Table 1 (rules (1)–(9)). Rule weights are governed by an underlying multinomial distribution and are shown in square brackets.

1. $S \rightarrow R(start)$	$[Pr = 1]$
2. $R(r_i.t) \rightarrow FS(r_j, start) R(r_j.t)$	$[P(r_j.t   r_i.t) \cdot \lambda]$
3. $R(r_i.t) \rightarrow FS(r_j, start)$	$[P(r_j.t   r_i.t) \cdot \lambda]$
4. $FS(r, r.f_i) \rightarrow F(r, r.f_j) FS(r, r.f_j)$	$[P(f_j   f_i)]$
5. $FS(r, r.f_i) \rightarrow F(r, r.f_j)$	$[P(f_j   f_i)]$
6. $F(r, r.f) \rightarrow W(r, r.f) F(r, r.f)$	$[P(w   w_{-1}, r, r.f)]$
7. $F(r, r.f) \rightarrow W(r, r.f)$	$[P(w   w_{-1}, r, r.f)]$
8. $W(r, r.f) \rightarrow \alpha$	$[P(\alpha   r, r.f, f.t, f.v)]$
9. $W(r, r.f) \rightarrow g(f.v)$	$[P(g(f.v).mode   r, r.f, f.t = int)]$

Table 1: Grammar rules and their weights shown in square brackets.

ets. Non-terminal symbols are in capitals and denote intermediate states; the terminal symbol  $\alpha$  corresponds to all words seen in the training set, and  $g(f.v)$  is a function for generating integer numbers given the value of a field  $f$ . All non-terminals, save the start symbol  $S$ , have one or more constraints (shown in parentheses), similar to number and gender agreement constraints in augmented syntactic rules.

Rule (1) denotes the expansion from the start symbol  $S$  to record  $R$ , which has the special *start* type (hence the notation  $R(start)$ ). Rule (2) defines a chain between two consecutive records  $r_i$  and  $r_j$ . Here,  $FS(r_j, start)$  represents the set of fields of the target  $r_j$ , following the source record  $R(r_i)$ . For example, the rule  $R(search_1.t) \rightarrow FS(flight_1, start)R(flight_1.t)$  can be interpreted as follows. Given that we have talked about *search*<sub>1</sub>, we will next talk about *flight*<sub>1</sub> and thus emit its corresponding fields.  $R(flight_1.t)$  is a non-terminal place-holder for the continuation of the chain of records, and *start* in  $FS$  is a special boundary field between consecutive records. The weight of this rule is the bigram probability of two records conditioned on their type, multiplied with a normalization factor  $\lambda$ . We have also defined a *null* record type i.e., a record that has no fields and acts as a smoother for words that may not correspond to a particular record. Rule (3) is simply an escape rule, so that the parsing process (on the record level) can finish.

Rule (4) is the equivalent of rule (2) at the field

level, i.e., it describes the chaining of two consecutive fields  $f_i$  and  $f_j$ . Non-terminal  $F(r,r,f)$  refers to field  $f$  of record  $r$ . For example, the rule  $FS(flight_1, from) \rightarrow F(flight_1, to)FS(flight_1, to)$ , specifies that we should talk about the field  $to$  of record  $flight_1$ , after talking about the field  $from$ . Analogously to the record level, we have also included a special *null* field type for the emission of words that do not correspond to a specific record field. Rule (6) defines the expansion of field  $F$  to a sequence of (binarized) words  $W$ , with a weight equal to the bigram probability of the current word given the previous word, the current record, and field.

Rules (8) and (9) define the emission of words and integer numbers from  $W$ , given a field type and its value. Rule (8) emits a single word from the vocabulary of the training set. Its weight defines a multinomial distribution over all seen words, for every value of field  $f$ , given that the field type is categorical or the special *null* field. Rule (9) is identical but for fields whose type is integer. Function  $g(f.v)$  generates an integer number given the field value, using either of the following six ways (Liang et al., 2009): identical to the field value, rounding up or rounding down to a multiple of 5, rounding off to the closest multiple of 5 and finally adding or subtracting some unexplained noise.<sup>1</sup> The weight is a multinomial over the six generation function *modes*, given the record field  $f$ .

The CFG in Table 1 will produce many derivations for a given input (i.e., a set of database records) which we represent compactly using a hypergraph or a packed forest (Klein and Manning, 2001; Huang, 2008). Simplified examples of this representation are shown in Figure 2.

### 3.2 Hypergraph Reranking

For our generation task, we are given a set of database records  $\mathbf{d}$ , and our goal is to find the best corresponding text  $\mathbf{w}$ . This corresponds to the best grammar derivation among a set of candidate derivations represented *implicitly* in the hypergraph structure. As shown in Table 1, the mapping from  $\mathbf{d}$  to  $\mathbf{w}$  is unknown. Therefore, all the intermediate multinomial distributions, described in the previous section, define a hidden correspondence structure  $\mathbf{h}$ , between records, fields, and their values. We find the best

<sup>1</sup>The noise is modeled as a geometric distribution.

---

#### Algorithm 1: Averaged Structured Perceptron

---

**Input:** Training scenarios:  $(\mathbf{d}_i, \mathbf{w}^*, \mathbf{h}_i^+)_{i=1}^N$   
1  $\alpha \leftarrow 0$   
2 **for**  $t \leftarrow 1 \dots T$  **do**  
3     **for**  $i \leftarrow 1 \dots N$  **do**  
4          $(\hat{\mathbf{w}}, \hat{\mathbf{h}}) = \arg \max_{w,h} \alpha \cdot \Phi(\mathbf{d}_i, \mathbf{w}_i, \mathbf{h}_i)$   
5         **if**  $(\mathbf{w}_i^*, \mathbf{h}_i^+) \neq (\hat{\mathbf{w}}_i, \hat{\mathbf{h}}_i)$  **then**  
6              $\alpha \leftarrow \alpha + \Phi(\mathbf{d}_i, \mathbf{w}_i^*, \mathbf{h}_i^+) - \Phi(\mathbf{d}_i, \hat{\mathbf{w}}_i, \hat{\mathbf{h}}_i)$   
7 **return**  $\frac{1}{T} \sum_{t=1}^T \frac{1}{N} \sum_{i=1}^N \alpha_t^i$

---

scoring derivation  $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$  by maximizing over configurations of  $\mathbf{h}$ :

$$(\hat{\mathbf{w}}, \hat{\mathbf{h}}) = \arg \max_{w,h} \alpha \cdot \Phi(\mathbf{d}, \mathbf{w}, \mathbf{h})$$

We define the score of  $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$  as the dot product between a high dimensional feature representation  $\Phi = (\Phi_1, \dots, \Phi_m)$  and a weight vector  $\alpha$ .

We estimate the weights  $\alpha$  using the averaged structured perceptron algorithm (Collins, 2002), which is well known for its speed and good performance in similar large-parameter NLP tasks (Liang et al., 2006; Huang, 2008). As shown in Algorithm 1, the perceptron makes several passes over the training scenarios, and in each iteration it computes the best scoring  $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$  among the candidate derivations, given the current weights  $\alpha$ . In line 6, the algorithm updates  $\alpha$  with the difference (if any) between the feature representations of the best scoring derivation  $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$  and the the *oracle derivation*  $(\mathbf{w}^*, \mathbf{h}^+)$ . Here,  $\hat{\mathbf{w}}$  is the estimated text,  $\mathbf{w}^*$  the gold-standard text,  $\hat{\mathbf{h}}$  is the estimated latent configuration of the model and  $\mathbf{h}^+$  the oracle latent configuration. The final weight vector  $\alpha$  is the average of weight vectors over  $T$  iterations and  $N$  scenarios. This averaging procedure avoids overfitting and produces more stable results (Collins, 2002).

In the following, we first explain how we decode in this framework, i.e., find the best scoring derivation (Section 3.3) and discuss our definition for the oracle derivation  $(\mathbf{w}^*, \mathbf{h}^+)$  (Section 3.4). Our features are described in Section 4.2.

### 3.3 Hypergraph Decoding

Following Huang (2008), we also distinguish features into local, i.e., those that can be computed within the confines of a single hyperedge, and non-local, i.e., those that require the prior visit of nodes other than their antecedents. For example, the



**Alignment** feature in Figure 2(a) is local, and thus can be computed a priori, but the **Word Trigrams** is not; in Figure 2(b) words in parentheses are sub-generations created so far at each word node; their combination gives rise to the trigrams serving as input to the feature. However, this combination may not take place at their immediate ancestors, since these may not be adjacent nodes in the hypergraph. According to the grammar in Table 1, there is no direct hyperedge between nodes representing words (W) and nodes representing the set of fields these correspond to (FS); rather, W and FS are connected implicitly via individual fields (F). Note, that in order to estimate the trigram feature at the FS node, we need to carry word information in the derivations of its antecedents, as we go bottom-up.<sup>2</sup>

Given these two types of features, we can then adapt Huang’s (2008) approximate decoding algorithm to find  $(\hat{\mathbf{w}}, \hat{\mathbf{h}})$ . Essentially, we perform bottom-up Viterbi search, visiting the nodes in reverse topological order, and keeping the  $k$ -best derivations for each. The score of each derivation is a linear combination of local and non-local features weights. In machine translation, a decoder that implements forest rescoring (Huang and Chiang, 2007) uses the language model as an external criterion of the goodness of sub-translations on account of their grammaticality. Analogously here, non-local features influence the selection of the best combinations, by introducing knowledge that exceeds the confines of the node under consideration and thus depend on the sub-derivations generated so far. (e.g., word trigrams spanning a field node rely on evidence from antecedent nodes that may be arbitrarily deeper than the field’s immediate children).

Our treatment of leaf nodes (see rules (8) and (9)) differs from the way these are usually handled in parsing. Since in generation we must emit rather than observe the words, for each leaf node we therefore output the  $k$ -best words according to the learned weights  $\alpha$  of the **Alignment** feature (see Section 4.2), and continue building our sub-generations bottom-up. This generation task is far from trivial: the search space on the word level is the size of the vocabulary and each field of a record can potentially generate all words. Also, note that in decoding it is useful to have a way to score different output

<sup>2</sup>We also store field information to compute structural features, described in Section 4.2.

lengths  $|\mathbf{w}|$ . Rather than setting  $\mathbf{w}$  to a fixed length, we rely on a linear regression predictor that uses the counts of each record type per scenario as features and is able to produce variable length texts.

### 3.4 Oracle Derivation

So far we have remained agnostic with respect to the oracle derivation  $(\mathbf{w}^*, \mathbf{h}^+)$ . In other NLP tasks such as syntactic parsing, there is a gold-standard parse, that can be used as the oracle. In our generation setting, such information is not available. We do not have the gold-standard alignment between the database records and the text that verbalizes them. Instead, we approximate it using the existing decoder to find the best latent configuration  $\mathbf{h}^+$  given the observed words in the training text  $\mathbf{w}^*$ .<sup>3</sup> This is similar in spirit to the generative alignment model of Liang et al. (2009).

## 4 Experimental Design

In this section we present our experimental setup for assessing the performance of our model. We give details on our dataset, model parameters and features, the approaches used for comparison, and explain how system output was evaluated.

### 4.1 Dataset

We conducted our experiments on the Air Travel Information System (ATIS) dataset (Dahl et al., 1994) which consists of transcriptions of spontaneous utterances of users interacting with a hypothetical on-line flight booking system. The dataset was originally created for the development of spoken language systems and is partitioned in individual user turns (e.g., *flights from orlando to milwaukee, show flights from orlando to milwaukee leaving after six o'clock*) each accompanied with an SQL query to a booking system and the results of this query. These utterances are typically short expressing a specific communicative goal (e.g., a question about the origin of a flight or its time of arrival). This inevitably results in small scenarios with a few words that often unambiguously correspond to a single record. To avoid training our model on a somewhat trivial corpus, we used the dataset introduced in Zettlemoyer

<sup>3</sup>In machine translation, Huang (2008) provides a soft algorithm that finds the forest oracle, i.e., the parse among the reranked candidates with the highest Parseval F-score. However, it still relies on the gold-standard reference translation.

and Collins (2007) instead, which combines the utterances of a single user in one scenario and contains 5,426 scenarios in total; each scenario corresponds to a (manually annotated) formal meaning representation ( $\lambda$ -expression) and its translation in natural language.

Lambda expressions were automatically converted into records, fields and values following the conventions adopted in Liang et al. (2009).<sup>4</sup> Given a lambda expression like the one shown in Figure 1, we first create a record for each variable and constant (e.g.,  $x$ , 9, august). We then assign record types according to the corresponding class types (e.g., variable  $x$  has class type *flight*). Next, fields and values are added from predicates with two arguments with the class type of the first argument matching that of the record type. The name of the predicate denotes the field, and the second argument denotes the value. We also defined special record types, such as *condition* and *search*. The latter is introduced for every lambda operator and assigned the categorical field *what* with the value *flight* which refers to the record type of variable  $x$ .

Contrary to datasets used in previous generation studies (e.g., ROBOCUP (Chen and Mooney, 2008) and WEATHERGOV (Liang et al., 2009)), ATIS has a much richer vocabulary (927 words); each scenario corresponds to a single sentence (average length is 11.2 words) with 2.65 out of 19 record types mentioned on average. Following Zettlemoyer and Collins (2007), we trained on 4,962 scenarios and tested on ATIS NOV93 which contains 448 examples.

## 4.2 Features

Broadly speaking, we defined two types of features, namely lexical and structural ones. In addition, we used a generatively trained PCFG as a baseline feature and an alignment feature based on the co-occurrence of records (or fields) with words.

**Baseline Feature** This is the log score of a generative decoder trained on the PCFG from Table 1. We converted the grammar into a hypergraph, and learned its probability distributions using a dynamic program similar to the inside-outside algorithm (Li and Eisner, 2009). Decoding was performed approx-

<sup>4</sup>The resulting dataset and a technical report describing the mapping procedure in detail are available from <http://homepages.inf.ed.ac.uk/s0793019/index.php?page=resources>

imately via cube pruning (Chiang, 2007), by integrating a trigram language model extracted from the training set (see Konstas and Lapata (2012) for details). Intuitively, the feature refers to the overall goodness of a specific derivation, applied locally in every hyperedge.

**Alignment Features** Instances of this feature family refer to the count of each PCFG rule from Table 1. For example, the number of times rule  $R(\text{search}_1.t) \rightarrow \text{FS}(\text{flight}_1, \text{start})R(\text{flight}_1.t)$  is included in a derivation (see Figure 2(a))

**Lexical Features** These features encourage grammatical coherence and inform lexical selection over and above the limited horizon of the language model captured by Rules (6)–(9). They also tackle anomalies in the generated output, due to the ergodicity of the CFG rules at the record and field level:

*Word Bigrams/Trigrams* This is a group of non-local feature functions that count word  $n$ -grams at every level in the hypergraph (see Figure 2(b)). The integration of words in the sub-derivations is adapted from Chiang (2007).

*Number of Words per Field* This feature function counts the number of words for every field, aiming to capture compound proper nouns and multi-word expressions, e.g., fields *from* and *to* frequently correspond to two or three words such as ‘*new york*’ and ‘*salt lake city*’ (see Figure 2(d)).

*Consecutive Word/Bigram/Trigram* This feature family targets adjacent repetitions of the same word, bigram or trigram, e.g., ‘*show me the show me the flights*’.

**Structural Features** Features in this category target primarily content selection and influence appropriate choice at the field level:

*Field bigrams/trigrams* Analogously to the lexical features mentioned above, we introduce a series of non-local features that capture field  $n$ -grams, given a specific record. For example the record *flight* in the air travel domain typically has the values  $\langle \text{from to} \rangle$  (see Figure 2(c)). The integration of fields in sub-derivations is implemented in fashion similar to the integration of words.

*Number of Fields per Record* This feature family is a coarser version of the *Field bigrams/trigrams*

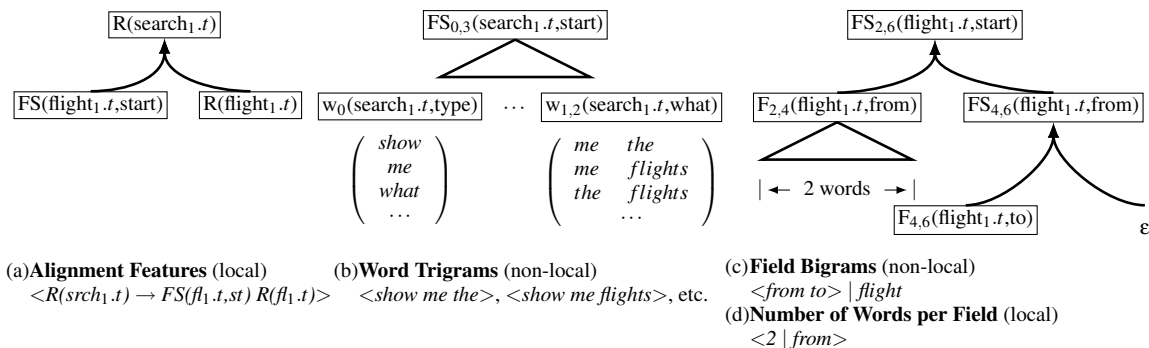


Figure 2: Simplified hypergraph examples with corresponding local and non-local features.

feature, which is deemed to be sparse for rarely-seen records.

*Field with No Value* Although records in the ATIS database schema have many fields, only a few are assigned a value in any given scenario. For example, the *flight* record has 13 fields, of which only 1.7 (on average) have a value. Practically, in a generative model this kind of sparsity would result in very low field recall. We thus include an identity feature function that explicitly counts whether a particular field has a value.

### 4.3 Evaluation

We evaluated three configurations of our model. A system that only uses the top scoring derivation in each sub-generation and incorporates only the baseline and alignment features (1-BEST+BASE+ALIGN). Our second system considers the  $k$ -best derivations and additionally includes lexical features ( $k$ -BEST+BASE+ALIGN+LEX). The number of  $k$ -best derivations was set to 40 and estimated experimentally on held-out data. And finally, our third system includes the full feature set ( $k$ -BEST+BASE+ALIGN+LEX+STR). Note, that the second and third system incorporate non-local features, hence the use of  $k$ -best derivation lists.<sup>5</sup> We compared our model to Angeli et al. (2010) whose approach is closest to ours.<sup>6</sup>

We evaluated system output automatically, using the BLEU-4 modified precision score (Papineni et

<sup>5</sup>Since the addition of these features, essentially incurs reranking, it follows that the systems would exhibit the exact same performance as the baseline system with 1-best lists.

<sup>6</sup>We are grateful to Gabor Angeli for providing us with the code of his system.

al., 2002) with the human-written text as reference. We also report results with the METEOR score (Banerjee and Lavie, 2005), which takes into account word re-ordering and has been shown to correlate better with human judgments at the sentence level. In addition, we evaluated the generated text by eliciting human judgments. Participants were presented with a scenario and its corresponding verbalization (see Figure 3) and were asked to rate the latter along two dimensions: fluency (is the text grammatical and overall understandable?) and semantic correctness (does the meaning conveyed by the text correspond to the database input?). The subjects used a five point rating scale where a high number indicates better performance. We randomly selected 12 documents from the test set and generated output with two of our models (1-BEST+BASE+ALIGN and  $k$ -BEST+BASE+ALIGN+LEX+STR) and Angeli et al.’s (2010) model. We also included the original text (HUMAN) as a gold standard. We thus obtained ratings for 48 ( $12 \times 4$ ) scenario-text pairs. The study was conducted over the Internet, using Amazon Mechanical Turk, and was completed by 51 volunteers, all self reported native English speakers.

## 5 Results

Table 2 summarizes our results. As can be seen, inclusion of lexical features gives our decoder an absolute increase of 6.73% in BLEU over the 1-BEST system. It also outperforms the discriminative system of Angeli et al. (2010). Our lexical features seem more robust compared to their templates. This is especially the case with infrequent records, where their system struggles to learn any meaningful information. Addition of the structural features further boosts performance. Our model increases by 8.69%

System	BLEU	METEOR
1-BEST+BASE+ALIGN	21.93	34.01
<i>k</i> -BEST+BASE+ALIGN+LEX	28.66	45.18
<i>k</i> -BEST+BASE+ALIGN+LEX+STR	30.62	46.07
ANGELI	26.77	42.41

Table 2: BLEU-4 and METEOR results on ATIS.

over the 1-BEST system and 3.85% over ANGELI in terms of BLEU. We observe a similar trend when evaluating system output with METEOR. Differences in magnitude are larger with the latter metric.

The results of our human evaluation study are shown in Table 5. We carried out an Analysis of Variance (ANOVA) to examine the effect of system type (1-BEST, *k*-BEST, ANGELI, and HUMAN) on the fluency and semantic correctness ratings. Means differences were compared using a post-hoc Tukey test. The *k*-BEST system is significantly better than the 1-BEST and ANGELI ( $a < 0.01$ ) both in terms of fluency and semantic correctness. ANGELI is significantly better than 1-BEST with regard to fluency ( $a < 0.05$ ) but not semantic correctness. There is no statistically significant difference between the *k*-BEST output and the original sentences (HUMAN).

Examples of system output are shown in Table 3. They broadly convey similar meaning with the gold-standard; ANGELI exhibits some long-range repetition, probably due to re-iteration of the same record patterns. We tackle this issue with the inclusion of non-local structural features. The 1-BEST system has some grammaticality issues, which we avoid by defining features over lexical *n*-grams and repeated words. It is worth noting that both our system and ANGELI produce output that is semantically compatible with but lexically different from the gold-standard (compare *please list the flights* and *show me the flights* against *give me the flights*). This is expected given the size of the vocabulary, but raises concerns regarding the use of automatic metrics for the evaluation of generation output.

## 6 Conclusions

We presented a discriminative reranking framework for an end-to-end generation system that performs both content selection and surface realization. Central to our approach is the encoding of generation as a parsing problem. We reformulate the input (a set of database records and text describing some of

System	Fluency	SemCor
1-BEST+BASE+ALIGN	2.70	3.05
<i>k</i> -BEST+BASE+ALIGN+LEX+STR	4.02	4.04
ANGELI	3.74	3.17
HUMAN	4.18	4.02

Table 3: Mean ratings for fluency and semantic correctness (SemCor) on system output elicited by humans.

	Flight	Time
	from to	when dep/ar
	phoenix milwaukee	evening departure
	Day	Search
	day dep/ar	type what
	wednesday departure	query flight
HUMAN	give me the flights from phoenix to milwaukee on wednesday evening	
ANGELI	show me the flights from phoenix to milwaukee on wednesday evening flights from phoenix to milwaukee	
<i>k</i> -BEST	please list the flights from phoenix to milwaukee on wednesday evening	
1-BEST	on wednesday evening from from phoenix to milwaukee on wednesday evening	

Figure 3: Example of scenario input and system output.

them) as a PCFG and convert it to a hypergraph. We find the best scoring derivation via forest reranking using both local and non-local features, that we train using the perceptron algorithm. Experimental evaluation on the ATIS dataset shows that our model attains significantly higher fluency and semantic correctness than any of the comparison systems. The current model can be easily extended to incorporate, additional, more elaborate features. Likewise, it can port to other domains with similar database structure without modification, such as WEATHERGOV and ROBOCUP. Finally, distributed training strategies have been developed for the perceptron algorithm (McDonald et al., 2010), which would allow our generator to scale to even larger datasets.

In the future, we would also like to tackle more challenging domains (e.g., product descriptions) and to enrich our generator with some notion of discourse planning. An interesting question is how to extend the PCFG-based approach advocated here so as to capture discourse-level document structure.

## References

- Gabor Angeli, Percy Liang, and Dan Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512, Cambridge, MA.
- Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 173–180, Ann Arbor, Michigan, June.
- David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of International Conference on Machine Learning*, pages 128–135, Helsinki, Finland.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Proceedings of the 17th International Conference on Machine Learning*, pages 175–182, Stanford, California.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8, Philadelphia, Pennsylvania.
- Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the ATIS task: the ATIS-3 corpus. In *Proceedings of the Workshop on Human Language Technology*, pages 43–48, Plainsboro, New Jersey.
- Robert Dale, Sabine Geldof, and Jean-Philippe Prost. 2003. Coral: Using natural language generation for navigational assistance. In *Proceedings of the 26th Australasian Computer Science Conference*, pages 35–44, Adelaide, Australia.
- Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. 1993. Directed hypergraphs and applications. *Discrete Applied Mathematics*, 42:177–201.
- Ruifang Ge and Raymond J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 263–270, Sydney, Australia.
- Nancy Green. 2006. Generation of biomedical arguments for lay readers. In *Proceedings of the 5th International Natural Language Generation Conference*, pages 114–121, Sydney, Australia.
- Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic.
- Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08: HLT*, pages 586–594, Columbus, Ohio.
- Dan Klein and Christopher D. Manning. 2001. Parsing and hypergraphs. In *Proceedings of the 7th International Workshop on Parsing Technologies*, pages 123–134, Beijing, China.
- Ioannis Konstas and Mirella Lapata. 2012. Unsupervised concept-to-text generation with hypergraphs. To appear in *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada.
- Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 40–51, Suntec, Singapore.
- Zhifei Li and Sanjeev Khudanpur. 2009. Forest reranking for machine translation with the perceptron algorithm. In *GALE Book*. GALE.
- Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 761–768, Sydney, Australia.
- Percy Liang, Michael Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, Suntec, Singapore.
- Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464, Los Angeles, CA, June. Association for Computational Linguistics.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of*

- the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania.
- Adwait Ratnaparkhi. 2002. Trainable approaches to surface natural language generation and their application to conversational dialog systems. *Computer Speech & Language*, 16(3-4):435–455.
- Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge University Press, New York, NY.
- Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167:137–169.
- Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *HLT-NAACL 2004: Main Proceedings*, pages 177–184, Boston, Massachusetts.
- Ross Turner, Yaji Sripada, and Ehud Reiter. 2009. Generating approximate geographic descriptions. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 42–49, Athens, Greece.
- Yuk Wah Wong and Raymond Mooney. 2007. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of the Human Language Technology and the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–179, Rochester, NY.
- Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 678–687, Prague, Czech Republic.

# A Discriminative Hierarchical Model for Fast Coreference at Large Scale

**Michael Wick**  
University of Massachusetts  
140 Governor’s Drive  
Amherst, MA  
mwick@cs.umass.edu

**Sameer Singh**  
University of Massachusetts  
140 Governor’s Drive  
Amherst, MA  
sameer@cs.umass.edu

**Andrew McCallum**  
University of Massachusetts  
140 Governor’s Drive  
Amherst, MA  
mccallum@cs.umass.edu

## Abstract

Methods that measure compatibility between mention pairs are currently the dominant approach to coreference. However, they suffer from a number of drawbacks including difficulties scaling to large numbers of mentions and limited representational power. As these drawbacks become increasingly restrictive, the need to replace the pairwise approaches with a more expressive, highly scalable alternative is becoming urgent. In this paper we propose a novel discriminative hierarchical model that recursively partitions entities into trees of latent sub-entities. These trees succinctly summarize the mentions providing a highly compact, information-rich structure for reasoning about entities and coreference uncertainty at massive scales. We demonstrate that the hierarchical model is several orders of magnitude faster than pairwise, allowing us to perform coreference on six million author mentions in under four hours on a single CPU.

## 1 Introduction

Coreference resolution, the task of clustering *mentions* into partitions representing their underlying real-world *entities*, is fundamental for high-level information extraction and data integration, including semantic search, question answering, and knowledge base construction. For example, coreference is vital for determining author publication lists in bibliographic knowledge bases such as CiteSeer and Google Scholar, where the repository must know if the “R. Hamming” who authored “Error detecting and error correcting codes” is the same” “R.

Hamming” who authored “The unreasonable effectiveness of mathematics.” Features of the mentions (e.g., bags-of-words in titles, contextual snippets and co-author lists) provide evidence for resolving such entities.

Over the years, various machine learning techniques have been applied to different variations of the coreference problem. A commonality in many of these approaches is that they model the problem of entity coreference as a collection of decisions between mention pairs (Bagga and Baldwin, 1999; Soon et al., 2001; McCallum and Wellner, 2004; Singla and Domingos, 2005; Bengston and Roth, 2008). That is, coreference is solved by answering a quadratic number of questions of the form “does *mention A* refer to the same entity as *mention B*?” with a compatibility function that indicates how likely A and B are coreferent. While these models have been successful in some domains, they also exhibit several undesirable characteristics. The first is that pairwise models lack the expressivity required to represent aggregate properties of the entities. Recent work has shown that these entity-level properties allow systems to correct coreference errors made from myopic pairwise decisions (Ng, 2005; Culotta et al., 2007; Yang et al., 2008; Rahman and Ng, 2009; Wick et al., 2009), and can even provide a strong signal for unsupervised coreference (Bhattacharya and Getoor, 2006; Haghighi and Klein, 2007; Haghighi and Klein, 2010).

A second problem, that has received significantly less attention in the literature, is that the pairwise coreference models scale poorly to large collections of mentions especially when the expected

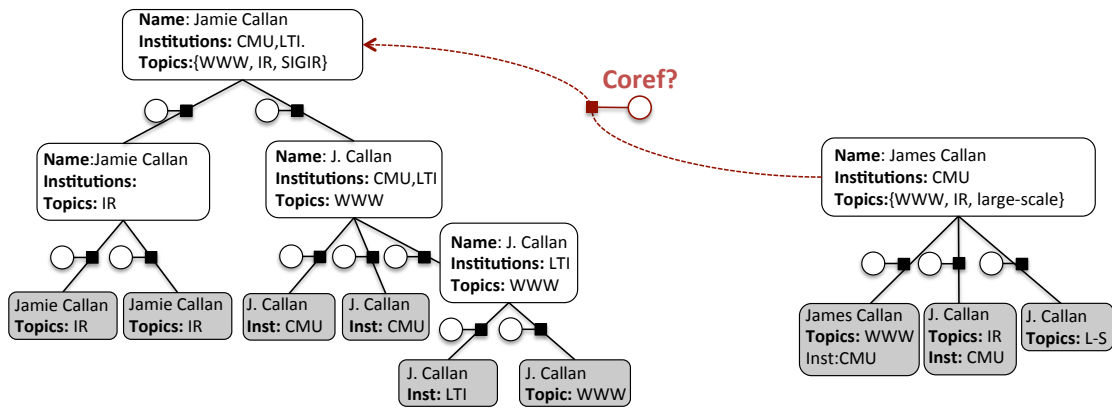


Figure 1: **Discriminative hierarchical factor graph for coreference:** Latent entity nodes (white boxes) summarize subtrees. Pairwise factors (black squares) measure compatibilities between child and parent nodes, avoiding quadratic blow-up. Corresponding decision variables (open circles) indicate whether one node is the child of another. Mentions (gray boxes) are leaves. Deciding whether to merge these two entities requires evaluating just a single factor (red square), corresponding to the new child-parent relationship.

number of mentions in each entity cluster is also large. Current systems cope with this by either dividing the data into blocks to reduce the search space (Hernández and Stolfo, 1995; McCallum et al., 2000; Bilenko et al., 2006), using fixed heuristics to greedily compress the mentions (Ravin and Kazi, 1999; Rao et al., 2010), employing specialized Markov chain Monte Carlo procedures (Milch et al., 2006; Richardson and Domingos, 2006; Singh et al., 2010), or introducing shallow hierarchies of sub-entities for MCMC block moves and super-entities for adaptive distributed inference (Singh et al., 2011). However, while these methods help manage the search space for medium-scale data, evaluating each coreference decision in many of these systems still scales linearly with the number of mentions in an entity, resulting in prohibitive computational costs associated with large datasets. This scaling with the number of mentions per entity seems particularly wasteful because although it is common for an entity to be referenced by a large number of mentions, many of these coreferent mentions are highly similar to each other. For example, in author coreference the two most common strings that refer to Richard Hamming might have the form “R. Hamming” and “Richard Hamming.” In newswire coreference, a prominent entity like Barack Obama may have millions of “Obama” mentions (many occurring in similar semantic contexts). Deciding whether

a mention belongs to this entity need not involve comparisons to all contextually similar “Obama” mentions; rather we prefer a more compact representation in order to efficiently reason about them.

In this paper we propose a novel hierarchical discriminative factor graph for coreference resolution that recursively structures each entity as a tree of latent sub-entities with mentions at the leaves. Our hierarchical model avoids the aforementioned problems of the pairwise approach: not only can it jointly reason about attributes of entire entities (using the power of discriminative conditional random fields), but it is also able to scale to datasets with enormous numbers of mentions because scoring entities does not require computing a quadratic number of compatibility functions. The key insight is that each node in the tree functions as a highly compact information-rich summary of its children. Thus, a small handful of upper-level nodes may summarize millions of mentions (for example, a single node may summarize all contextually similar “R. Hamming” mentions). Although inferring the structure of the entities requires reasoning over a larger state-space, the latent trees are actually beneficial to inference (as shown for shallow trees in Singh et al. (2011)), resulting in rapid progress toward high probability regions, and mirroring known benefits of auxiliary variable methods in statistical physics (such as Swendsen and Wang (1987)). Moreover,



each step of inference is computationally efficient because evaluating the cost of attaching (or detaching) sub-trees requires computing just a single compatibility function (as seen in Figure 1). Further, our hierarchical approach provides a number of additional advantages. First, the recursive nature of the tree (arbitrary depth and width) allows the model to adapt to different types of data and effectively compress entities of different scales (e.g., entities with more mentions may require a deeper hierarchy to compress). Second, the model contains compatibility functions at all levels of the tree enabling it to simultaneously reason at multiple granularities of entity compression. Third, the trees can provide split points for finer-grained entities by placing contextually similar mentions under the same subtree. Finally, if memory is limited, redundant mentions can be pruned by replacing subtrees with their roots.

Empirically, we demonstrate that our model is several orders of magnitude faster than a pairwise model, allowing us to perform efficient coreference on nearly six million author mentions in under four hours using a single CPU.

## 2 Background: Pairwise Coreference

Coreference is the problem of clustering mentions such that mentions in the same set refer to the same real-world entity; it is also known as entity disambiguation, record linkage, and de-duplication. For example, in author coreference, each mention might be represented as a record extracted from the author field of a textual citation or BibTeX record. The mention record may contain attributes for the first, middle, and last name of the author, as well as contextual information occurring in the citation string, co-authors, titles, topics, and institutions. The goal is to cluster these mention records into sets, each containing all the mentions of the author to which they refer; we use this task as a running pedagogical example.

Let  $\mathcal{M}$  be the space of observed mention records; then the traditional pairwise coreference approach scores candidate coreference solutions with a compatibility function  $\psi : \mathcal{M} \times \mathcal{M} \rightarrow \mathfrak{R}$  that measures how likely it is that the two mentions refer to the same entity.<sup>1</sup> In discriminative log-

<sup>1</sup>We can also include an *incompatibility* function for when

linear models, the function  $\psi$  takes the form of weights  $\theta$  on features  $\phi(m_i, m_j)$ , i.e.,  $\psi(m_i, m_j) = \exp(\theta \cdot \phi(m_i, m_j))$ . For example, in author coreference, the feature functions  $\phi$  might test whether the name fields for two author mentions are string identical, or compute cosine similarity between the two mentions' bags-of-words, each representing a mention's context. The corresponding real-valued weights  $\theta$  determine the impact of these features on the overall pairwise score.

Coreference can be solved by introducing a set of binary coreference decision variables for each mention pair and predicting a setting to their values that maximizes the sum of pairwise compatibility functions. While it is possible to independently make pairwise decisions and enforce transitivity *post hoc*, this can lead to poor accuracy because the decisions are tightly coupled. For higher accuracy, a graphical model such as a conditional random field (CRF) is constructed from the compatibility functions to jointly reason about the pairwise decisions (McCallum and Wellner, 2004). We now describe the pairwise CRF for coreference as a factor graph.

### 2.1 Pairwise Conditional Random Field

Each mention  $m_i \in \mathcal{M}$  is an observed variable, and for each mention pair  $(m_i, m_j)$  we have a binary coreference decision variable  $y_{ij}$  whose value determines whether  $m_i$  and  $m_j$  refer to the same entity (i.e., 1 means they are coreferent and 0 means they are not coreferent). The pairwise compatibility functions become the factors in the graphical model. Each factor examines the properties of its mention pair as well as the setting to the coreference decision variable and outputs a score indicating how likely the setting of that coreference variable is. The joint probability distribution over all possible settings to the coreference decision variables ( $\mathbf{y}$ ) is given as a product of all the pairwise compatibility factors:

$$Pr(\mathbf{y}|\mathbf{m}) \propto \prod_{i=1}^n \prod_{j=1}^n \psi(m_i, m_j, y_{ij}) \quad (1)$$

Given the pairwise CRF, the problem of coreference is then solved by searching for the setting of the coreference decision variables that has the highest probability according to Equation 1 subject to the mentions are not coreferent, e.g.,  $\psi : \mathcal{M} \times \mathcal{M} \times \{0, 1\} \rightarrow \mathfrak{R}$

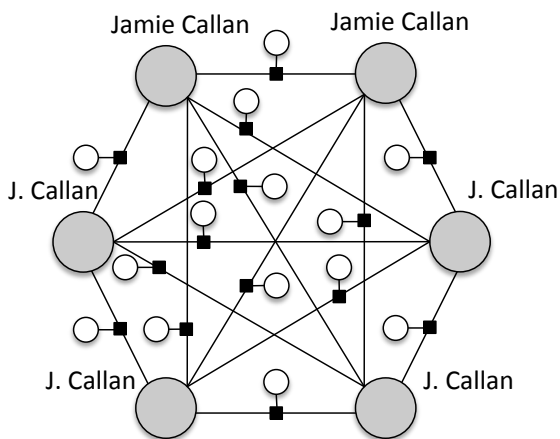


Figure 2: **Pairwise model on six mentions:** Open circles are the binary coreference decision variables, shaded circles are the observed mentions, and the black boxes are the factors of the graphical model that encode the pairwise compatibility functions.

constraint that the setting to the coreference variables obey transitivity;<sup>2</sup> this is the maximum probability estimate (MPE) setting. However, the solution to this problem is intractable, and even approximate inference methods such as loopy belief propagation can be difficult due to the cubic number of deterministic transitivity constraints.

## 2.2 Approximate Inference

An approximate inference framework that has successfully been used for coreference models is Metropolis-Hastings (MH) (Milch et al. (2006), Culotta and McCallum (2006), Poon and Domingos (2007), amongst others), a Markov chain Monte Carlo algorithm traditionally used for marginal inference, but which can also be tuned for MPE inference. MH is a flexible framework for specifying customized local-search transition functions and provides a principled way of deciding which local search moves to accept. A proposal function  $q$  takes the current coreference hypothesis and proposes a new hypothesis by modifying a subset of the decision variables. The proposed change is accepted with probability  $\alpha$ :

$$\alpha = \min \left( 1, \frac{Pr(\mathbf{y}') q(\mathbf{y}|\mathbf{y}')}{Pr(\mathbf{y}) q(\mathbf{y}'|\mathbf{y})} \right) \quad (2)$$

<sup>2</sup>We say that a full assignment to the coreference variables  $\mathbf{y}$  obeys transitivity if  $\forall ijk y_{ij} = 1 \wedge y_{jk} = 1 \implies y_{ik} = 1$

When using MH for MPE inference, the second term  $q(\mathbf{y}|\mathbf{y}')/q(\mathbf{y}'|\mathbf{y})$  is optional, and usually omitted. Moves that reduce model score may be accepted and an optional temperature can be used for annealing. The primary advantages of MH for coreference are (1) only the compatibility functions of the changed decision variables need to be evaluated to accept a move, and (2) the proposal function can enforce the transitivity constraint by exploring only variable settings that result in valid coreference partitionings.

A commonly used proposal distribution for coreference is the following: (1) randomly select two mentions  $(m_i, m_j)$ , (2) if the mentions  $(m_i, m_j)$  are in the same entity cluster according to  $\mathbf{y}$  then move one mention into a singleton cluster (by setting the necessary decision variables to 0), otherwise, move mention  $m_i$  so it is in the same cluster as  $m_j$  (by setting the necessary decision variables). Typically, MH is employed by first initializing to a singleton configuration (all entities have one mention), and then executing the MH for a certain number of steps (or until the predicted coreference hypothesis stops changing).

This proposal distribution always moves a single mention  $m$  from some entity  $e_i$  to another entity  $e_j$  and thus the configuration  $\mathbf{y}$  and  $\mathbf{y}'$  only differ by the setting of decision variables governing to which entity  $m$  refers. In order to guarantee transitivity and a valid coreference equivalence relation, we must properly remove  $m$  from  $e_i$  by untethering  $m$  from each mention in  $e_i$  (this requires computing  $|e_i| - 1$  pairwise factors). Similarly—again, for the sake of transitivity—in order to complete the move into  $e_j$  we must coref  $m$  to each mention in  $e_j$  (this requires computing  $|e_j|$  pairwise factors). Clearly, all the other coreference decision variables are independent and so their corresponding factors cancel because they yield the same scores under  $\mathbf{y}$  and  $\mathbf{y}'$ . Thus, evaluating each proposal for the pairwise model scales linearly with the number of mentions assigned to the entities, requiring the evaluation of  $2(|e_i| + |e_j| - 1)$  compatibility functions (factors).

## 3 Hierarchical Coreference

Instead of only capturing a single coreference clustering between mention pairs, we can imagine multiple levels of coreference decisions over different

granularities. For example, mentions of an author may be further partitioned into semantically similar sets, such that mentions from each set have topically similar papers. This partitioning can be recursive, i.e., each of these sets can be further partitioned, capturing candidate splits for an entity that can facilitate inference. In this section, we describe a model that captures arbitrarily deep hierarchies over such layers of coreference decisions, enabling efficient inference and rich entity representations.

### 3.1 Discriminative Hierarchical Model

In contrast to the pairwise model, where each entity is a flat cluster of mentions, our proposed model structures each entity recursively as a tree. The leaves of the tree are the observed mentions with a set of attribute values. Each internal node of the tree is latent and contains a set of unobserved attributes; recursively, these *node records* summarize the attributes of their child nodes (see Figure 1), for example, they may aggregate the bags of context words of the children. The root of each tree represents the entire entity, with the leaves containing its mentions. Formally, the coreference decision variables in the hierarchical model no longer represent pairwise decisions directly. Instead, a decision variable  $y_{r_i, r_j} = 1$  indicates that node-record  $r_j$  is the parent of node-record  $r_i$ . We say a node-record *exists* if either it is a mention, has a parent, or has at least one child. Let  $R$  be the set of all existing node records, let  $r^p$  denote the parent for node  $r$ , that is  $y_{r, r^p} = 1$ , and  $\forall r' \neq r^p, y_{r, r'} = 0$ . As we describe in more detail later, the structure of the tree and the values of the unobserved attributes are determined during inference.

In order to represent our recursive model of coreference, we include two types of factors: pairwise factors  $\psi_{pw}$  that measure compatibility between a child node-record and its parent, and unit-wise factors  $\psi_{rw}$  that measure compatibilities of the node-records themselves. For efficiency we enforce that parent-child factors only produce a non-zero score when the corresponding decision variable is 1. The unit-wise factors can examine compatibility of settings to the attribute variables for a particular node (for example, the set of topics may be too diverse to represent just a single entity), as well as enforce priors over the tree’s breadth and depth. Our recur-

sive hierarchical model defines the probability of a configuration as:

$$Pr(\mathbf{y}, R | \mathbf{m}) \propto \prod_{r \in R} \psi_{rw}(r) \psi_{pw}(r, r^p) \quad (3)$$

### 3.2 MCMC Inference for Hierarchical models

The state space of our hierarchical model is substantially larger (theoretically infinite) than the pairwise model due to the arbitrarily deep (and wide) latent structure of the cluster trees. Inference must simultaneously determine the structure of the tree, the latent node-record values, as well as the coreference decisions themselves.

While this may seem daunting, the structures being inferred are actually beneficial to inference. Indeed, despite the enlarged state space, inference in the hierarchical model is substantially faster than a pairwise model with a smaller state space. One explanatory intuition comes from the statistical physics community: we can view the latent tree as auxiliary variables in a data-augmentation sampling scheme that guide MCMC through the state space more efficiently. There is a large body of literature in the statistics community describing how these auxiliary variables can lead to faster convergence despite the enlarged state space (classic examples include Swendsen and Wang (1987) and slice samplers (Neal, 2000)).

Further, evaluating each proposal during inference in the hierarchical model is substantially faster than in the pairwise model. Indeed, we can replace the linear number of factor evaluations (as in the pairwise model) with a constant number of factor evaluations for most proposals (for example, adding a subtree requires re-evaluating only a single parent-child factor between the subtree and the attachment point, and a single node-wise factor).

Since inference must determine the structure of the entity trees in addition to coreference, it is advantageous to consider multiple MH proposals per sample. Therefore, we employ a modified variant of MH that is similar to multi-try Metropolis (Liu et al., 2000). Our modified MH algorithm makes  $k$  proposals and samples one according to its model ratio score (the first term in Equation 2) normalized across all  $k$ . More specifically, for each MH step, we first randomly select two subtrees headed by node-

records  $r_i$  and  $r_j$  from the current coreference hypothesis. If  $r_i$  and  $r_j$  are in different clusters, we propose several alternate merge operations: (also in Figure 3):

- **Merge Left** - merges the entire subtree of  $r_j$  into node  $r_i$  by making  $r_j$  a child of  $r_i$
- **Merge Entity Left** - merges  $r_j$  with  $r_i$ 's root
- **Merge Left and Collapse** - merges  $r_j$  into  $r_i$  then performs a collapse on  $r_j$  (see below).
- **Merge Up** - merges node  $r_i$  with node  $r_j$  by creating a new parent node-record variable  $r^p$  with  $r_i$  and  $r_j$  as the children. The attribute fields of  $r^p$  are selected from  $r_i$  and  $r_j$ .

Otherwise  $r_i$  and  $r_j$  are subtrees in the same entity tree, then the following proposals are used instead:

- **Split Right** - Make the subtree  $r_j$  the root of a new entity by detaching it from its parent
- **Collapse** - If  $r_i$  has a parent, then move  $r_i$ 's children to  $r_i$ 's parent and then delete  $r_i$ .
- **Sample attribute** - Pick a new value for an attribute of  $r_i$  from its children.

Computing the model ratio for all of coreference proposals requires only a constant number of compatibility functions. On the other hand, evaluating proposals in the pairwise model requires evaluating a number of compatibility functions equal to the number of mentions in the clusters being modified.

Note that changes to the attribute values of the node-record and collapsing still require evaluating a linear number of factors, but this is only linear in the number of child nodes, not linear in the number of mentions referring to the entity. Further, attribute values rarely change once the entities stabilize. Finally, we incrementally update bags during coreference to reflect the aggregates of their children.

## 4 Experiments: Author Coreference

Author coreference is a tremendously important task, enabling improved search and mining of scientific papers by researchers, funding agencies, and governments. The problem is extremely difficult due to the wide variations of names, limited contextual evidence, misspellings, people with common names, lack of standard citation formats, and large numbers of mentions.

For this task we use a publicly available collection of 4,394 BibTeX files containing 817,193 en-

tries.<sup>3</sup> We extract 1,322,985 author mentions, each containing first, middle, last names, bags-of-words of paper titles, topics in paper titles (by running latent Dirichlet allocation (Blei et al., 2003)), and last names of co-authors. In addition we include 2,833 mentions from the REXA dataset<sup>4</sup> labeled for coreference, in order to assess accuracy. We also include  $\sim 5$  million mentions from DBLP.

### 4.1 Models and Inference

Due to the paucity of labeled training data, we did not estimate parameters from data, but rather set the compatibility functions manually by specifying their log scores. The pairwise compatibility functions punish a string difference in first, middle, and last name, ( $-8$ ); reward a match ( $+2$ ); and reward matching initials ( $+1$ ). Additionally, we use the cosine similarity (shifted and scaled between  $-4$  and  $4$ ) between the bags-of-words containing title tokens, topics, and co-author last names. These compatibility functions define the scores of the factors in the pairwise model and the parent-child factors in the hierarchical model. Additionally, we include priors over the model structure. We encourage each node to have eight children using a per node factor having score  $1/(|\text{number of children}-8|+1)$ , manage tree depth by placing a cost on the creation of intermediate tree nodes  $-8$  and encourage clustering by placing a cost on the creation of root-level entities  $-7$ . These weights were determined by just a few hours of tuning on a development set.

We initialize the MCMC procedures to the singleton configuration (each entity consists of one mention) for each model, and run the MH algorithm described in Section 2.2 for the pairwise model and multi-try MH (described in Section 3.2) for the hierarchical model. We augment these samplers using canopies constructed by concatenating the first initial and last name: that is, mentions are only selected from within the same canopy (or block) to reduce the search space (Bilenko et al., 2006). During the course of MCMC inference, we record the pairwise F1 scores of the labeled subset. The source code for our model is available as part of the FACTORIE package (McCallum et al., 2009, <http://www.iesl.cs.umass.edu/data/bibtex>

<sup>3</sup><http://www.iesl.cs.umass.edu/data/bibtex>

<sup>4</sup><http://www2.selu.edu/Academics/Faculty/aculotta/data/rexa.html>

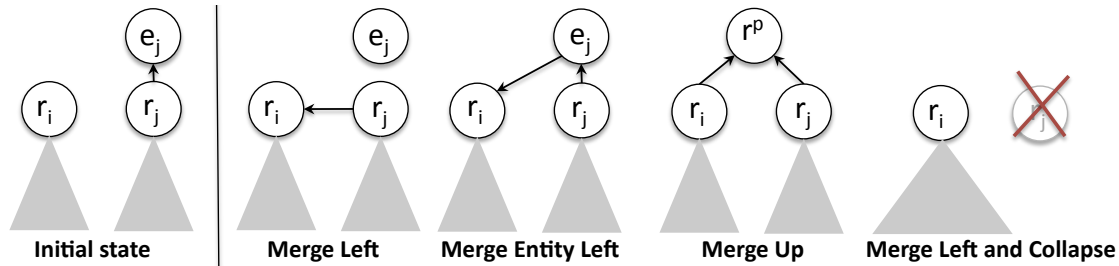


Figure 3: Example coreference proposals for the case where  $r_i$  and  $r_j$  are initially in different clusters.

//factorie.cs.umass.edu/).

## 4.2 Comparison to Pairwise Model

In Figure 4a we plot the number of samples completed over time for a 145k subset of the data. Recall that we initialized to the singleton configuration and that as the size of the entities grows, the cost of evaluating the entities in MCMC becomes more expensive. The pairwise model struggles with the large cluster sizes while the hierarchical model is hardly affected. Even though the hierarchical model is evaluating up to four proposals for each sample, it is still able to sample much faster than the pairwise model; this is expected because the cost of evaluating a proposal requires evaluating fewer factors. Next, we plot coreference F1 accuracy over time and show in Figure 5a that the prolific sampling rate of the hierarchical model results in faster coreference. Using the plot, we can compare running times for any desired level of accuracy. For example, on the 145k mention dataset, at a 60% accuracy level the hierarchical model is 19 times faster and at 90% accuracy it is 31 times faster. These performance improvements are even more profound on larger datasets: the hierarchical model achieves a 60% level of accuracy 72 times faster than the pairwise model on the 1.3 million mention dataset, reaching 90% in just 2,350 seconds. Note, however, that the hierarchical model requires more samples to reach a similar level of accuracy due to the larger state space (Figure 4b).

## 4.3 Large Scale Experiments

In order to demonstrate the scalability of the hierarchical model, we run it on nearly 5 million author mentions from DBLP. In under two hours (6,700 seconds), we achieve an accuracy of 80%, and in under three hours (10,600 seconds), we achieve an

accuracy of over 90%. Finally, we combine DBLP with BibTeX data to produce a dataset with almost 6 million mentions (5,803,811). Our performance on this dataset is similar to DBLP, taking just 13,500 seconds to reach a 90% accuracy.

## 5 Related Work

Singh et al. (2011) introduce a hierarchical model for coreference that treats entities as a two-tiered structure, by introducing the concept of sub-entities and super-entities. Super-entities reduce the search space in order to propose fruitful jumps. Sub-entities provide a tighter granularity of coreference and can be used to perform larger block moves during MCMC. However, the hierarchy is fixed and shallow. In contrast, our model can be arbitrarily deep and wide. Even more importantly, their model has pairwise factors and suffers from the quadratic curse, which they address by distributing inference.

The work of Rao et al. (2010) uses streaming clustering for large-scale coreference. However, the greedy nature of the approach does not allow errors to be revisited. Further, they compress entities by averaging their mentions' features. We are able to provide richer entity compression, the ability to revisit errors, and scale to larger data.

Our hierarchical model provides the advantages of recently proposed entity-based coreference systems that are known to provide higher accuracy (Haghighi and Klein, 2007; Culotta et al., 2007; Yang et al., 2008; Wick et al., 2009; Haghighi and Klein, 2010). However, these systems reason over a single layer of entities and do not scale well.

Techniques such as lifted inference (Singla and Domingos, 2008) for graphical models exploit redundancy in the data, but typically do not achieve any significant compression on coreference data be-

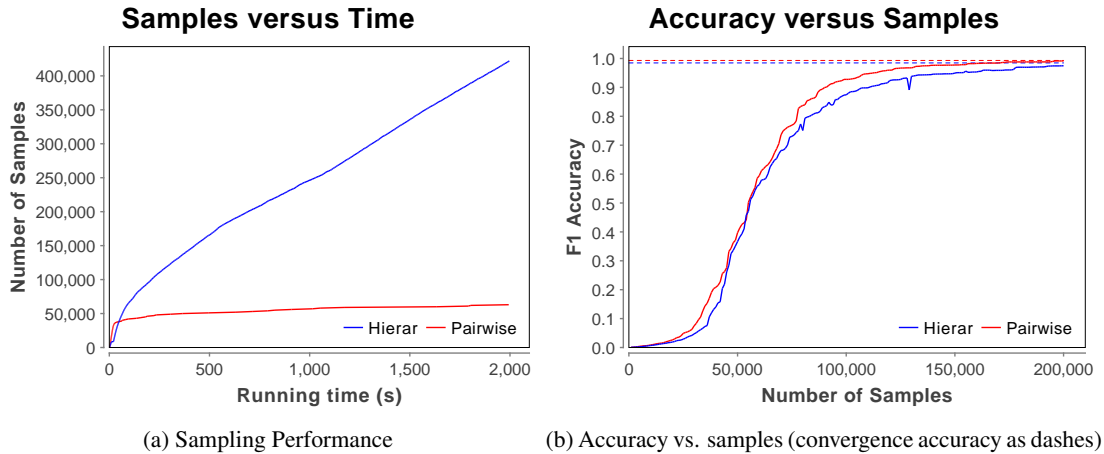


Figure 4: Sampling Performance Plots for 145k mentions

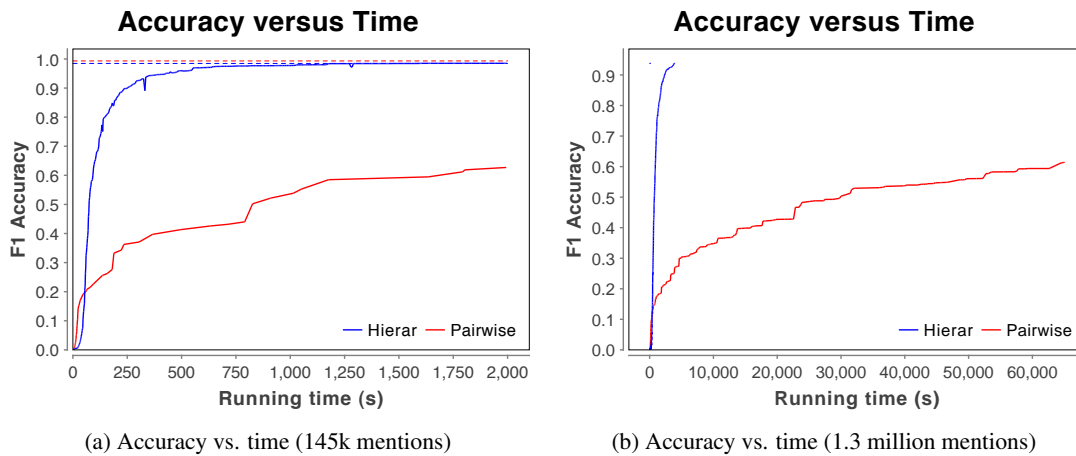


Figure 5: Runtime performance on two datasets

cause the observations usually violate any symmetry assumptions. On the other hand, our model is able to compress similar (but potentially different) observations together in order to make inference fast even in the presence of asymmetric observed data.

## 6 Conclusion

In this paper we present a new hierarchical model for large scale coreference and demonstrate it on the problem of author disambiguation. Our model recursively defines an entity as a summary of its children nodes, allowing succinct representations of millions of mentions. Indeed, inference in the hierarchy is orders of magnitude faster than a pairwise CRF, allowing us to infer accurate coreference on

six million mentions on one CPU in just 4 hours.

## 7 Acknowledgments

We would like to thank Veselin Stoyanov for his feedback. This work was supported in part by the CIIR, in part by ARFL under prime contract #FA8650-10-C-7059, in part by DARPA under AFRL prime contract #FA8750-09-C-0181, and in part by IARPA via DoI/NBC contract #D11PC20152. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

## References

- Amit Bagga and Breck Baldwin. 1999. Cross-document event coreference: annotations, experiments, and observations. In *Proceedings of the Workshop on Coreference and its Applications*, CorefApp '99, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Eric Bengston and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Empirical Methods in Natural Language Processing (EMNLP)*.
- Indrajit Bhattacharya and Lise Getoor. 2006. A latent Dirichlet model for unsupervised entity resolution. In *SDM*.
- Mikhail Bilenko, Beena Kamath, and Raymond J. Mooney. 2006. Adaptive blocking: Learning to scale up record linkage. In *Proceedings of the Sixth International Conference on Data Mining, ICDM '06*, pages 87–96, Washington, DC, USA. IEEE Computer Society.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal on Machine Learning Research*, 3:993–1022.
- Aron Culotta and Andrew McCallum. 2006. Practical Markov logic containing first-order quantifiers with application to identity uncertainty. In *Human Language Technology Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing (HLT/NAACL)*, June.
- Aron Culotta, Michael Wick, and Andrew McCallum. 2007. First-order probabilistic models for coreference resolution. In *North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT)*.
- Aria Haghighi and Dan Klein. 2007. Unsupervised coreference resolution in a nonparametric bayesian model. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 848–855.
- Aria Haghighi and Dan Klein. 2010. Coreference resolution in a modular, entity-centered model. In *North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT)*, pages 385–393.
- Mauricio A. Hernández and Salvatore J. Stolfo. 1995. The merge/purge problem for large databases. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of data, SIGMOD '95*, pages 127–138, New York, NY, USA. ACM.
- Jun S. Liu, Faming Liang, and Wing Hung Wong. 2000. The multiple-try method and local optimization in metropolis sampling. *Journal of the American Statistical Association*, 96(449):121–134.
- Andrew McCallum and Ben Wellner. 2004. Conditional models of identity uncertainty with application to noun coreference. In *Neural Information Processing Systems (NIPS)*.
- Andrew McCallum, Kamal Nigam, and Lyle Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 169–178.
- Andrew McCallum, Karl Schultz, and Sameer Singh. 2009. FACTORIE: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems (NIPS)*.
- Brian Milch, Bhaskara Marthi, and Stuart Russell. 2006. *BLOG: Relational Modeling with Unknown Objects*. Ph.D. thesis, University of California, Berkeley.
- Radford Neal. 2000. Slice sampling. *Annals of Statistics*, 31:705–767.
- Vincent Ng. 2005. Machine learning for coreference resolution: From local classification to global ranking. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Hoifung Poon and Pedro Domingos. 2007. Joint inference in information extraction. In *AAAI Conference on Artificial Intelligence*, pages 913–918.
- Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2, EMNLP '09*, pages 968–977, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Delip Rao, Paul McNamee, and Mark Dredze. 2010. Streaming cross document entity coreference resolution. In *International Conference on Computational Linguistics (COLING)*, pages 1050–1058, Beijing, China, August. Coling 2010 Organizing Committee.
- Yael Ravin and Zunaid Kazi. 1999. Is Hillary Rodham Clinton the president? disambiguating names across documents. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 9–16.
- Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning*, 62(1-2):107–136.
- Sameer Singh, Michael L. Wick, and Andrew McCallum. 2010. Distantly labeling data for large scale cross-document coreference. *Computing Research Repository (CoRR)*, abs/1005.4298.
- Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Association for Computational Linguistics: Human Language Technologies (ACL HLT)*.

- Parag Singla and Pedro Domingos. 2005. Discriminative training of Markov logic networks. In *AAAI*, Pittsburgh, PA.
- Parag Singla and Pedro Domingos. 2008. Lifted first-order belief propagation. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, pages 1094–1099. AAAI Press.
- Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Comput. Linguist.*, 27(4):521–544.
- R.H. Swendsen and J.S. Wang. 1987. Nonuniversal critical dynamics in MC simulations. *Phys. Rev. Lett.*, 58(2):68–88.
- Michael Wick, Aron Culotta, Khashayar Rohanimanesh, and Andrew McCallum. 2009. An entity-based model for coreference resolution. In *SIAM International Conference on Data Mining (SDM)*.
- 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 *Association for Computational Linguistics*, pages 843–851.



# Coreference Semantics from Web Features

Mohit Bansal and Dan Klein

Computer Science Division

University of California, Berkeley

{mbansal, klein}@cs.berkeley.edu

## Abstract

To address semantic ambiguities in coreference resolution, we use Web  $n$ -gram features that capture a range of world knowledge in a diffuse but robust way. Specifically, we exploit short-distance cues to hypernymy, semantic compatibility, and semantic context, as well as general lexical co-occurrence. When added to a state-of-the-art coreference baseline, our Web features give significant gains on multiple datasets (ACE 2004 and ACE 2005) and metrics (MUC and B<sup>3</sup>), resulting in the best results reported to date for the end-to-end task of coreference resolution.

## 1 Introduction

Many of the most difficult ambiguities in coreference resolution are semantic in nature. For instance, consider the following example:

*When Obama met Jobs, the president discussed the economy, technology, and education. His election campaign is expected to [...]*

For resolving coreference in this example, a system would benefit from the world knowledge that *Obama is the president*. Also, to resolve the pronoun *his* to the correct antecedent *Obama*, we can use the knowledge that *Obama* has an *election campaign* while *Jobs* does not. Such ambiguities are difficult to resolve on purely syntactic or configurational grounds.

There have been multiple previous systems that incorporate some form of world knowledge in coreference resolution tasks. Most work (Poesio et al., 2004; Markert and Nissim, 2005; Yang et al., 2005; Bergsma and Lin, 2006) addresses special cases and subtasks such as bridging anaphora,

other anaphora, definite NP reference, and pronoun resolution, computing semantic compatibility via Web-hits and counts from large corpora. There is also work on end-to-end coreference resolution that uses large noun-similarity lists (Daumé III and Marcu, 2005) or structured knowledge bases such as Wikipedia (Yang and Su, 2007; Haghighi and Klein, 2009; Kobdani et al., 2011) and YAGO (Rahman and Ng, 2011). However, such structured knowledge bases are of limited scope, and, while Haghighi and Klein (2010) self-acquires knowledge about coreference, it does so only via reference constructions and on a limited scale.

In this paper, we look to the Web for broader if shallower sources of semantics. In order to harness the information on the Web without presupposing a deep understanding of all Web text, we instead turn to a diverse collection of Web  $n$ -gram counts (Brants and Franz, 2006) which, in aggregate, contain diffuse and indirect, but often robust, cues to reference. For example, we can collect the co-occurrence statistics of an anaphor with various candidate antecedents to judge relative surface affinities (i.e., (*Obama, president*) versus (*Jobs, president*)). We can also count co-occurrence statistics of competing antecedents when placed in the context of an anaphoric pronoun (i.e., *Obama's election campaign* versus *Jobs' election campaign*).

All of our features begin with a pair of headwords from candidate mention pairs and compute statistics derived from various potentially informative queries' counts. We explore five major categories of semantically informative Web features, based on (1) general lexical affinities (via generic co-occurrence statistics), (2) lexical relations (via Hearst-style hypernymy patterns), (3) similarity of entity-based context (e.g., common values of  $y$  for

which *h is a y* is attested), (4) matches of distributional soft cluster ids, and (5) attested substitutions of candidate antecedents in the context of a pronominal anaphor.

We first describe a strong baseline consisting of the mention-pair model of the Reconcile system (Stoyanov et al., 2009; Stoyanov et al., 2010) using a decision tree (DT) as its pairwise classifier. To this baseline system, we add our suite of features in turn, each class of features providing substantial gains. Altogether, our final system produces the best numbers reported to date on end-to-end coreference resolution (with automatically detected system mentions) on multiple data sets (ACE 2004 and ACE 2005) and metrics (MUC and B<sup>3</sup>), achieving significant improvements over the Reconcile DT baseline and over the state-of-the-art results of Haghighi and Klein (2010).

## 2 Baseline System

Before describing our semantic Web features, we first describe our baseline. The core inference and features come from the Reconcile package (Stoyanov et al., 2009; Stoyanov et al., 2010), with modifications described below. Our baseline differs most substantially from Stoyanov et al. (2009) in using a decision tree classifier rather than an averaged linear perceptron.

### 2.1 Reconcile

Reconcile is one of the best implementations of the *mention-pair* model (Soon et al., 2001) of coreference resolution. The mention-pair model relies on a pairwise function to determine whether or not two mentions are coreferent. Pairwise predictions are then consolidated by transitive closure (or some other clustering method) to form the final set of coreference clusters (chains). While our Web features could be adapted to entity-mention systems, their current form was most directly applicable to the mention-pair approach, making Reconcile a particularly well-suited platform for this investigation.

The Reconcile system provides baseline features, learning mechanisms, and resolution procedures that already achieve near state-of-the-art results on multiple popular datasets using multiple standard metrics. It includes over 80 core features that exploit

various automatically generated annotations such as named entity tags, syntactic parses, and WordNet classes, inspired by Soon et al. (2001), Ng and Cardie (2002), and Bengtson and Roth (2008). The Reconcile system also facilitates standardized empirical evaluation to past work.<sup>1</sup>

In this paper, we develop a suite of simple semantic Web features based on pairs of mention headwords which stack with the default Reconcile features to surpass past state-of-the-art results.

### 2.2 Decision Tree Classifier

Among the various learning algorithms that Reconcile supports, we chose the decision tree classifier, available in Weka (Hall et al., 2009) as J48, an open source Java implementation of the C4.5 algorithm of Quinlan (1993).

The C4.5 algorithm builds decision trees by incrementally maximizing information gain. The training data is a set of already classified samples, where each sample is a vector of attributes or features. At each node of the tree, C4.5 splits the data on an attribute that most effectively splits its set of samples into more ordered subsets, and then recurses on these smaller subsets. The decision tree can then be used to classify a new sample by following a path from the root downward based on the attribute values of the sample.

We find the decision tree classifier to work better than the default averaged perceptron (used by Stoyanov et al. (2009)), on multiple datasets using multiple metrics (see Section 4.3). Many advantages have been claimed for decision tree classifiers, including interpretability and robustness. However, we suspect that the aspect most relevant to our case is that decision trees can capture non-linear interactions between features. For example, recency is very important for pronoun reference but much less so for nominal reference.

## 3 Semantics via Web Features

Our Web features for coreference resolution are simple and capture a range of diffuse world knowledge. Given a mention pair, we use the head finder in Reconcile to find the lexical heads of both mentions (for

---

<sup>1</sup>We use the default configuration settings of Reconcile (Stoyanov et al., 2010) unless mentioned otherwise.

example, the head of *the Palestinian territories* is the word *territories*). Next, we take each headword pair ( $h_1, h_2$ ) and compute various Web-count functions on it that can signal whether or not this mention pair is coreferent.

As the source of Web information, we use the Google  $n$ -grams corpus (Brants and Franz, 2006) which contains English  $n$ -grams ( $n = 1$  to 5) and their Web frequency counts, derived from nearly 1 trillion word tokens and 95 billion sentences. Because we have many queries that must be run against this corpus, we apply the trie-based hashing algorithm of Bansal and Klein (2011) to efficiently answer all of them in one pass over it. The features that require word clusters (Section 3.4) use the output of Lin et al. (2010).<sup>2</sup>

We describe our five types of features in turn. The first four types are most intuitive for mention pairs where both members are non-pronominal, but, aside from the general co-occurrence group, helped for all mention pair types. The fifth feature group applies only to pairs in which the anaphor is a pronoun but the antecedent is a non-pronoun. Related work for each feature category is discussed inline.

### 3.1 General co-occurrence

These features capture co-occurrence statistics of the two headwords, i.e., how often  $h_1$  and  $h_2$  are seen adjacent or nearly adjacent on the Web. This count can be a useful coreference signal because, in general, mentions referring to the same entity will co-occur more frequently (in large corpora) than those that do not. Using the  $n$ -grams corpus (for  $n = 1$  to 5), we collect co-occurrence Web-counts by allowing a varying number of wildcards between  $h_1$  and  $h_2$  in the query. The co-occurrence value is:

$$\text{bin} \left( \log_{10} \left( \frac{c_{12}}{c_1 \cdot c_2} \right) \right)$$

<sup>2</sup>These clusters are derived from the V2 Google  $n$ -grams corpus. The V2 corpus itself is not publicly available, but the clusters are available at <http://www.clsp.jhu.edu/~sbergma/PhrasalClusters>

where

$$\begin{aligned} c_{12} &= \text{count}("h_1 \star h_2") \\ &+ \text{count}("h_1 \star \star h_2") \\ &+ \text{count}("h_1 \star \star \star h_2"), \\ c_1 &= \text{count}("h_1"), \text{ and} \\ c_2 &= \text{count}("h_2"). \end{aligned}$$

We normalize the overall co-occurrence count of the headword pair  $c_{12}$  by the unigram counts of the individual headwords  $c_1$  and  $c_2$ , so that high-frequency headwords do not unfairly get a high feature value (this is similar to computing scaled mutual information MI (Church and Hanks, 1989)).<sup>3</sup> This normalized value is quantized by taking its  $\log_{10}$  and binning. The actual feature that fires is an indicator of which quantized bin the query produced. As a real example from our development set, the co-occurrence count  $c_{12}$  for the headword pair (*leader, president*) is 11383, while it is only 95 for the headword pair (*voter, president*); after normalization and  $\log_{10}$ , the values are -10.9 and -12.0, respectively.

These kinds of general Web co-occurrence statistics have been used previously for other supervised NLP tasks such as spelling correction and syntactic parsing (Bergsma et al., 2010; Bansal and Klein, 2011). In coreference, similar word-association scores were used by Kobdani et al. (2011), but from Wikipedia and for self-training.

### 3.2 Hearst co-occurrence

These features capture templated co-occurrence of the two headwords  $h_1$  and  $h_2$  in the Web-corpus. Here, we only collect statistics of the headwords co-occurring with a generalized Hearst pattern (Hearst, 1992) in between. Hearst patterns capture various lexical semantic relations between items. For example, seeing *X is a Y* or *X and other Y* indicates hypernymy and also tends to cue coreference. The specific patterns we use are:

- $h_1 \{is \mid are \mid was \mid were\} \{a \mid an \mid the\} h_2$
- $h_1 \{and \mid or\} \{other \mid the \text{ other} \mid another\} h_2$
- $h_1 \text{ other than } \{a \mid an \mid the\} h_2$

<sup>3</sup>We also tried adding  $\text{count}("h_1 h_2")$  to  $c_{12}$  but this decreases performance, perhaps because truly adjacent occurrences are often not grammatical.

- $h_1$  such as  $\{a \mid an \mid the\}?$   $h_2$
- $h_1$ , including  $\{a \mid an \mid the\}?$   $h_2$
- $h_1$ , especially  $\{a \mid an \mid the\}?$   $h_2$
- $h_1$  of  $\{the \mid all\}?$   $h_2$

For this feature, we again use a quantized normalized count as in Section 3.1, but  $c_{12}$  here is restricted to  $n$ -grams where one of the above patterns occurs in between the headwords. We did not allow wildcards in between the headwords and the Hearst-patterns because this introduced a significant amount of noise. Also, we do not constrain the order of  $h_1$  and  $h_2$  because these patterns can hold for either direction of coreference.<sup>4</sup> As a real example from our development set, the  $c_{12}$  count for the headword pair (*leader*, *president*) is 752, while for (*voter*, *president*), it is 0.

Hypernymic semantic compatibility for coreference is intuitive and has been explored in varying forms by previous work. Poesio et al. (2004) and Markert and Nissim (2005) employ a subset of our Hearst patterns and Web-hits for the subtasks of bridging anaphora, other-anaphora, and definite NP resolution. Others (Haghighi and Klein, 2009; Rahman and Ng, 2011; Daumé III and Marcu, 2005) use similar relations to extract compatibility statistics from Wikipedia, YAGO, and noun-similarity lists. Yang and Su (2007) use Wikipedia to automatically extract semantic patterns, which are then used as features in a learning setup. Instead of extracting patterns from the training data, we use all the above patterns, which helps us generalize to new datasets for end-to-end coreference resolution (see Section 4.3).

### 3.3 Entity-based context

For each headword  $h$ , we first collect context *seeds*  $y$  using the pattern

$$h \{is \mid are \mid was \mid were\} \{a \mid an \mid the\}?$$

taking seeds  $y$  in order of decreasing Web count. The corresponding ordered *seed list*  $Y = \{y\}$  gives us useful information about the headword’s entity type. For example, for  $h = \textit{president}$ , the top

<sup>4</sup>Two minor variants not listed above are  $h_1$  including  $h_2$  and  $h_1$  especially  $h_2$ .

30 seeds (and their parts of speech) include important cues such as *president is elected* (verb), *president is authorized* (verb), *president is responsible* (adjective), *president is the chief* (adjective), *president is above* (preposition), and *president is the head* (noun).

Matches in the seed lists of two headwords can be a strong signal that they are coreferent. For example, in the top 30 seed lists for the headword pair (*leader*, *president*), we get matches including *elected*, *responsible*, and *expected*. To capture this effect, we create a feature that indicates whether there is a match in the top  $k$  seeds of the two headwords (where  $k$  is a hyperparameter to tune).

We create another feature that indicates whether the dominant parts of speech in the seed lists matches for the headword pair. We first collect the POS tags (using length 2 character prefixes to indicate coarse parts of speech) of the seeds matched in the top  $k'$  seed lists of the two headwords, where  $k'$  is another hyperparameter to tune. If the dominant tags match and are in a small list of important tags ( $\{JJ, NN, RB, VB\}$ ), we fire an indicator feature specifying the matched tag, otherwise we fire a *no-match* indicator. To obtain POS tags for the seeds, we use a unigram-based POS tagger trained on the WSJ treebank training set.

### 3.4 Cluster information

The distributional hypothesis of Harris (1954) says that words that occur in similar contexts tend to have a similar linguistic behavior. Here, we design features with the idea that this hypothesis extends to reference: mentions occurring in similar contexts in large document sets such as the Web tend to be compatible for coreference. Instead of collecting the contexts of each mention and creating sparse features from them, we use Web-scale distributional clustering to summarize compatibility.

Specifically, we begin with the phrase-based clusters from Lin et al. (2010), which were created using the Google  $n$ -grams V2 corpus. These clusters come from distributional  $K$ -Means clustering (with  $K = 1000$ ) on phrases, using the  $n$ -gram context as features. The cluster data contains almost 10 million phrases and their *soft* cluster memberships. Up to twenty cluster ids with the highest centroid similarities are included for each phrase in this dataset

(Lin et al., 2010).

Our cluster-based features assume that if the headwords of the two mentions have matches in their cluster id lists, then they are more compatible for coreference. We check the match of not just the top 1 cluster ids, but also farther down in the 20 sized lists because, as discussed in Lin and Wu (2009), the soft cluster assignments often reveal different senses of a word. However, we also assume that higher-ranked matches tend to imply closer meanings. To this end, we fire a feature indicating the value  $\text{bin}(i+j)$ , where  $i$  and  $j$  are the *earliest match* positions in the cluster id lists of  $h_1$  and  $h_2$ . Binning here means that match positions in a close range generally trigger the same feature.

Recent previous work has used clustering information to improve the performance of supervised NLP tasks such as NER and dependency parsing (Koo et al., 2008; Lin and Wu, 2009). However, in coreference, the only related work to our knowledge is from Daumé III and Marcu (2005), who use word class features derived from a Web-scale corpus via a process described in Ravichandran et al. (2005).

### 3.5 Pronoun context

Our last feature category specifically addresses pronoun reference, for cases when the anaphoric mention  $NP_2$  (and hence its headword  $h_2$ ) is a pronoun, while the candidate antecedent mention  $NP_1$  (and hence its headword  $h_1$ ) is not. For such a headword pair ( $h_1, h_2$ ), the idea is to substitute the non-pronoun  $h_1$  into  $h_2$ 's position and see whether the result is attested on the Web.

If the anaphoric pronominal mention is  $h_2$  and its sentential context is  $l' l h_2 r r'$ , then the substituted phrase will be  $l' l h_1 r r'$ .<sup>5</sup> High Web counts of substituted phrases tend to indicate semantic compatibility. Perhaps unsurprisingly for English, only the right context was useful in this capacity. We chose the following three context types, based on performance on a development set:

<sup>5</sup>Possessive pronouns are replaced with an additional apostrophe, i.e.,  $h_1 's$ . We also use features (see R1Gap) that allow wildcards ( $\star$ ) in between the headword and the context when collecting Web-counts, in order to allow for determiners and other filler words.

$$\bullet h_1 r \quad (\text{R1})$$

$$\bullet h_1 r r' \quad (\text{R2})$$

$$\bullet h_1 \star r \quad (\text{R1Gap})$$

As an example of the R1Gap feature, if the anaphor  $h_2$  + context is *his victory* and one candidate antecedent  $h_1$  is *Bush*, then we compute the normalized value

$$\frac{\text{count}(\text{"Bush 's } \star \text{ victory"})}{\text{count}(\text{" } \star \text{ 's } \star \text{ victory"})\text{count}(\text{"Bush"})}$$

In general, we compute

$$\frac{\text{count}(\text{"}h_1 \text{'s } \star \text{ r"})}{\text{count}(\text{" } \star \text{'s } \star \text{ r"})\text{count}(\text{"}h_1 \text{"})}$$

The final feature value is again a normalized count converted to  $\log_{10}$  and then binned.<sup>6</sup> We have three separate features for the R1, R2, and R1Gap context types. We tune a separate bin-size hyperparameter for each of these three features.

These pronoun resolution features are similar to selectional preference work by Yang et al. (2005) and Bergsma and Lin (2006), who compute semantic compatibility for pronouns in specific syntactic relationships such as possessive-noun, subject-verb, etc. In our case, we directly use the general context of any pronominal anaphor to find its most compatible antecedent.

Note that all our above features are designed to be non-sparse by firing indicators of the quantized Web statistics and not the lexical- or class-based identities of the mention pair. This keeps the total number of features small, which is important for the relatively small datasets used for coreference resolution. We go from around 100 features in the Reconcile baseline to around 165 features after adding all our Web features.

<sup>6</sup>Normalization helps us with two kinds of balancing. First, we divide by the count of the antecedent so that when choosing the best antecedent for a fixed anaphor, we are not biased towards more frequently occurring antecedents. Second, we divide by the count of the context so that across anaphora, an anaphor with rarer context does not get smaller values (for all its candidate antecedents) than another anaphor with a more common context.

Dataset	docs	dev	test	ment	chn
ACE04	128	63/27	90/38	3037	1332
ACE05	81	40/17	57/24	1991	775
ACE05-ALL	599	337/145	482/117	9217	3050

Table 1: Dataset characteristics – *docs*: the total number of documents; *dev*: the train/test split during development; *test*: the train/test split during testing; *ment*: the number of gold mentions in the test split; *chn*: the number of coreference chains in the test split.

## 4 Experiments

### 4.1 Data

We show results on three popular and comparatively larger coreference resolution data sets – the ACE04, ACE05, and ACE05-ALL datasets from the ACE Program (NIST, 2004). In ACE04 and ACE05, we have only the newswire portion (of the original ACE 2004 and 2005 training sets) and use the standard train/test splits reported in Stoyanov et al. (2009) and Haghighi and Klein (2010). In ACE05-ALL, we have the full ACE 2005 training set and use the standard train/test splits reported in Rahman and Ng (2009) and Haghighi and Klein (2010). Note that most previous work does not report (or need) a standard development set; hence, for tuning our features and its hyper-parameters, we randomly split the original training data into a training and development set with a 70/30 ratio (and then use the full original training set during testing). Details of the corpora are shown in Table 1.<sup>7</sup>

Details of the Web-scale corpora used for extracting features are discussed in Section 3.

### 4.2 Evaluation Metrics

We evaluated our work on both MUC (Vilain et al., 1995) and  $B^3$  (Bagga and Baldwin, 1998). Both scorers are available in the Reconcile infrastructure.<sup>8</sup> MUC measures how many predicted clusters need to be merged to cover the true gold clusters.  $B^3$  computes precision and recall for each mention by computing the intersection of its predicted and gold cluster and dividing by the size of the predicted

<sup>7</sup>Note that the development set is used only for ACE04, because for ACE05, and ACE05-ALL, we directly test using the features tuned on ACE04.

<sup>8</sup>Note that  $B^3$  has two versions which handle twinless (spurious) mentions in different ways (see Stoyanov et al. (2009) for details). We use the  $B^3$ All version, unless mentioned otherwise.

Feature	MUC			$B^3$		
	P	R	F1	P	R	F1
AvgPerc	69.0	63.1	65.9	82.2	69.9	75.5
DecTree	<b>80.9</b>	61.0	69.5	<b>89.5</b>	69.0	77.9
+ Co-occ	79.8	62.1	69.8	88.7	69.8	78.1
+ Hearst	80.0	62.3	70.0	89.1	70.1	78.5
+ Entity	79.4	63.2	70.4	88.1	70.9	78.6
+ Cluster	79.5	63.6	70.7	87.9	71.2	78.6
+ Pronoun	79.9	<b>64.3</b>	<b>71.3</b>	88.0	<b>71.6</b>	<b>79.0</b>

Table 2: Incremental results for the Web features on the ACE04 development set. AvgPerc is the averaged perceptron baseline, DecTree is the decision tree baseline, and the +Feature rows show the effect of adding a particular feature incrementally (not in isolation) to the DecTree baseline. The feature categories correspond to those described in Section 3.

and gold cluster, respectively. It is well known (Recasens and Hovy, 2010; Ng, 2010; Kobdani et al., 2011) that MUC is biased towards large clusters (chains) whereas  $B^3$  is biased towards singleton clusters. Therefore, for a more balanced evaluation, we show improvements on both metrics simultaneously.

### 4.3 Results

We start with the Reconcile baseline but employ the decision tree (DT) classifier, because it has significantly better performance than the default averaged perceptron classifier used in Stoyanov et al. (2009).<sup>9</sup> Table 2 compares the baseline perceptron results to the DT results and then shows the incremental addition of the Web features to the DT baseline (on the ACE04 development set).

The DT classifier, in general, is precision-biased. The Web features somewhat balance this by increasing the recall and decreasing precision to a lesser extent, improving overall F1. Each feature type incrementally increases both MUC and  $B^3$  F1-measures, showing that they are not taking advantage of any bias of either metric. The incremental improvements also show that each Web feature type brings in some additional benefit over the information already present in the Reconcile baseline, which includes alias, animacy, named entity, and WordNet

<sup>9</sup>Moreover, a DT classifier takes roughly the same amount of time and memory as a perceptron on our ACE04 development experiments. It is, however, slower and more memory-intensive ( $\sim 3x$ ) on the bigger ACE05-ALL dataset.

System	MUC			B <sup>3</sup>		
	P	R	F1	P	R	F1
<b>ACE04-TEST-RESULTS</b>						
Stoyanov et al. (2009)	-	-	62.0	-	-	76.5
Haghighi and Klein (2009)	67.5	61.6	64.4	77.4	69.4	73.2
Haghighi and Klein (2010)	67.4	<b>66.6</b>	67.0	81.2	73.3	77.0
This Work: Perceptron Baseline	65.5	61.9	63.7	84.1	70.9	77.0
This Work: DT Baseline	<b>76.0</b>	60.7	67.5	<b>89.6</b>	70.3	78.8
This Work: DT + Web Features	74.8	64.2	<b>69.1</b>	87.5	<b>73.7</b>	<b>80.0</b>
This Work: $\Delta$ of DT+Web over DT	$(p < 0.05)$		1.7	$(p < 0.005)$		1.3
<b>ACE05-TEST-RESULTS</b>						
Stoyanov et al. (2009)	-	-	67.4	-	-	73.7
Haghighi and Klein (2009)	73.1	58.8	65.2	82.1	63.9	71.8
Haghighi and Klein (2010)	74.6	62.7	68.1	83.2	68.4	75.1
This Work: Perceptron Baseline	72.2	61.6	66.5	85.0	65.5	73.9
This Work: DT Baseline	<b>79.6</b>	59.7	68.2	<b>89.4</b>	64.2	74.7
This Work: DT + Web Features	75.0	<b>64.7</b>	<b>69.5</b>	81.1	<b>70.8</b>	<b>75.6</b>
This Work: $\Delta$ of DT+Web over DT	$(p < 0.12)$		1.3	$(p < 0.1)$		0.9
<b>ACE05-ALL-TEST-RESULTS</b>						
Rahman and Ng (2009)	75.4	64.1	69.3	54.4	70.5	61.4
Haghighi and Klein (2009)	72.9	60.2	67.0	53.2	73.1	61.6
Haghighi and Klein (2010)	77.0	<b>66.9</b>	<b>71.6</b>	55.4	<b>74.8</b>	63.8
This Work: Perceptron Baseline	68.9	60.4	64.4	80.6	60.5	69.1
This Work: DT Baseline	<b>78.0</b>	60.4	68.1	<b>85.1</b>	60.4	70.6
This Work: DT + Web Features	77.6	64.0	70.2	80.7	65.9	<b>72.5</b>
This Work: $\Delta$ of DT+Web over DT	$(p < 0.001)$		2.1	$(p < 0.001)$		1.9

Table 3: Primary test results on the ACE04, ACE05, and ACE05-ALL datasets. All systems reported here use automatically extracted system mentions. B<sup>3</sup> here is the B<sup>3</sup>All version of Stoyanov et al. (2009). We also report statistical significance of the improvements from the Web features on the DT baseline, using the bootstrap test (Noreen, 1989; Efron and Tibshirani, 1993). The perceptron baseline in this work (Reconcile settings: 15 iterations, threshold = 0.45, SIG for ACE04 and AP for ACE05, ACE05-ALL) has different results from Stoyanov et al. (2009) because their current publicly available code is different from that used in their paper (*p.c.*). Also, the B<sup>3</sup> variant used by Rahman and Ng (2009) is slightly different from other systems (they remove all and only the singleton twinless system mentions, so it is neither B<sup>3</sup>All nor B<sup>3</sup>None). For completeness, our (untuned) B<sup>3</sup>None results (DT + Web) on the ACE05-ALL dataset are P=69.9|R=65.9|F1=67.8.

class / sense information.<sup>10</sup>

Table 3 shows our primary *test* results on the ACE04, ACE05, and ACE05-ALL datasets, for the MUC and B<sup>3</sup> metrics. All systems reported use automatically detected mentions. We report our results (the 3 rows marked ‘This Work’) on the perceptron baseline, the DT baseline, and the Web features added to the DT baseline. We also report statistical significance of the improvements from the Web fea-

<sup>10</sup>We also initially experimented with smaller datasets (MUC6 and MUC7) and an averaged perceptron baseline, and we did see similar improvements, arguing that these features are useful independently of the learning algorithm and dataset.

tures on the DT baseline.<sup>11</sup> For significance testing, we use the bootstrap test (Noreen, 1989; Efron and Tibshirani, 1993).

Our main comparison is against Haghighi and Klein (2010), a mostly-unsupervised generative approach that models latent entity types, which generate specific entities that in turn render individual mentions. They learn on large datasets including

<sup>11</sup>All improvements are significant, except on the small ACE05 dataset with the MUC metric (where it is weak, at  $p < 0.12$ ). However, on the larger version of this dataset, ACE05-ALL, we get improvements which are both larger and more significant (at  $p < 0.001$ ).

Wikipedia, and their results are state-of-the-art in coreference resolution. We outperform their system on most datasets and metrics (except on ACE05-ALL for the MUC metric). The other systems we compare to and outperform are the perceptron-based Reconcile system of Stoyanov et al. (2009), the strong deterministic system of Haghighi and Klein (2009), and the cluster-ranking model of Rahman and Ng (2009).

We develop our features and tune their hyperparameter values on the ACE04 development set and then use these on the ACE04 test set.<sup>12</sup> On the ACE05 and ACE05-ALL datasets, we directly transfer our Web features and their hyper-parameter values from the ACE04 dev-set, without any retuning. The test improvements we get on all the datasets (see Table 3) suggest that our features are generally useful across datasets and metrics.<sup>13</sup>

## 5 Analysis

In this section, we briefly discuss errors (in the DT baseline) corrected by our Web features, and analyze the decision tree classifier built during training (based on the ACE04 development experiments).

To study error correction, we begin with the mention pairs that are coreferent according to the gold-standard annotation (after matching the system mentions to the gold ones). We consider the pairs that are wrongly predicted to be non-coreferent by the baseline DT system but correctly predicted to be coreferent when we add our Web features. Some examples of such pairs include:

*Iran ; the country*  
*the EPA ; the agency*  
*athletic director ; Mulcahy*  
*Democrat Al Gore ; the vice president*

---

<sup>12</sup>Note that for the ACE04 dataset only, we use the ‘SmartInstanceGenerator’ (SIG) filter of Reconcile that uses only a filtered set of mention-pairs (based on distance and other properties of the pair) instead of the ‘AllPairs’ (AP) setting that uses all pairs of mentions, and makes training and tuning very slow.

<sup>13</sup>For the ACE05 and ACE05-ALL datasets, we revert to the ‘AllPairs’ (AP) setting of Reconcile because this gives us baselines competitive with Haghighi and Klein (2010). Since we did not need to retune on these datasets, training and tuning speed were not a bottleneck. Moreover, the improvements from our Web features are similar even when tried over the SIG baseline; hence, the filter choice doesn’t affect the performance gain from the Web features.

*Barry Bonds ; the best baseball player*  
*Vojislav Kostunica ; the pro-democracy leader*  
*its closest rival ; the German magazine Das Motorrad*  
*One of those difficult-to-dislodge judges ; John Marshall*

These pairs are cases where our features on Hearst-style co-occurrence and entity-based context-match are informative and help discriminate in favor of the correct antecedents. One advantage of using Web-based features is that the Web has a surprising amount of information on even rare entities such as proper names. Our features also correct coreference for various cases of pronominal anaphora, but these corrections are harder to convey out of context.

Next, we analyze the decision tree built after training the classifier (with all our Web features included). Around 30% of the decision nodes (both non-terminals and leaves) correspond to Web features, and the average error in classification at the Web-feature leaves is only around 2.5%, suggesting that our features are strongly discriminative for pairwise coreference decisions. Some of the most discriminative nodes correspond to the general co-occurrence feature for most (binned) log-count values, the Hearst-style co-occurrence feature for its zero-count value, the cluster-match feature for its zero-match value, and the R2 pronoun context feature for certain (binned) log-count values.

## 6 Conclusion

We have presented a collection of simple Web-count features for coreference resolution that capture a range of world knowledge via statistics of general lexical co-occurrence, hypernymy, semantic compatibility, and semantic context. When added to a strong decision tree baseline, these features give significant improvements and achieve the best results reported to date, across multiple datasets and metrics.

## Acknowledgments

We would like to thank Nathan Gilbert, Adam Pauls, and the anonymous reviewers for their helpful suggestions. This research is supported by Qualcomm via an Innovation Fellowship to the first author and by BBN under DARPA contract HR0011-12-C-0014.



## References

- Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of MUC-7 and LREC Workshop*.
- Mohit Bansal and Dan Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of ACL*.
- Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of EMNLP*.
- Shane Bergsma and Dekang Lin. 2006. Bootstrapping path-based pronoun resolution. In *Proceedings of COLING-ACL*.
- Shane Bergsma, Emily Pitler, and Dekang Lin. 2010. Creating robust supervised classifiers via web-scale n-gram data. In *Proceedings of ACL*.
- Thorsten Brants and Alex Franz. 2006. The Google Web 1T 5-gram corpus version 1.1. *LDC2006T13*.
- Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *Proceedings of ACL*.
- Hal Daumé III and Daniel Marcu. 2005. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proceedings of EMNLP*.
- B. Efron and R. Tibshirani. 1993. An introduction to the bootstrap. *Chapman & Hall CRC*.
- Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of EMNLP*.
- Aria Haghighi and Dan Klein. 2010. Coreference resolution in a modular, entity-centered model. In *Proceedings of NAACL-HLT*.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1).
- Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146162.
- Marti Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING*.
- Hamidreza Kobdani, Hinrich Schutze, Michael Schiehlen, and Hans Kamp. 2011. Bootstrapping coreference resolution using word associations. In *Proceedings of ACL*.
- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*.
- Dekang Lin and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of ACL*.
- Dekang Lin, Kenneth Church, Heng Ji, Satoshi Sekine, David Yarowsky, Shane Bergsma, Kailash Patil, Emily Pitler, Rachel Lathbury, Vikram Rao, Kapil Dalwani, and Sushant Narsale. 2010. New tools for web-scale n-grams. In *Proceedings of LREC*.
- Katja Markert and Malvina Nissim. 2005. Comparing knowledge sources for nominal anaphora resolution. *Computational Linguistics*, 31(3):367–402.
- Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of ACL*.
- Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of ACL*.
- NIST. 2004. The ACE evaluation plan. In *NIST*.
- E.W. Noreen. 1989. Computer intensive methods for hypothesis testing: An introduction. *Wiley, New York*.
- Massimo Poesio, Rahul Mehta, Axel Maroudas, and Janet Hitzeman. 2004. Learning to resolve bridging references. In *Proceedings of ACL*.
- J. R. Quinlan. 1993. C4.5: Programs for machine learning. *Morgan Kaufmann Publishers Inc., San Francisco, CA, USA*.
- Altaf Rahman and Vincent Ng. 2009. Supervised models for coreference resolution. In *Proceedings of EMNLP*.
- Altaf Rahman and Vincent Ng. 2011. Coreference resolution with world knowledge. In *Proceedings of ACL*.
- Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and NLP: Using locality sensitive hash functions for high speed noun clustering. In *Proceedings of ACL*.
- Marta Recasens and Eduard Hovy. 2010. Coreference resolution across corpora: Languages, coding schemes, and preprocessing information. In *Proceedings of ACL*.
- Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.
- Veselin Stoyanov, Nathan Gilbert, Claire Cardie, and Ellen Riloff. 2009. Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of ACL/IJCNLP*.
- Veselin Stoyanov, Claire Cardie, Nathan Gilbert, Ellen Riloff, David Buttler, and David Hysom. 2010. Reconcile: A coreference resolution research platform. In *Technical report, Cornell University*.
- Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of MUC-6*.
- Xiaofeng Yang and Jian Su. 2007. Coreference resolution using semantic relatedness information from automatically discovered patterns. In *Proceedings of ACL*.

Xiaofeng Yang, Jian Su, and Chew Lim Tan. 2005. Improving pronoun resolution using statistics-based semantic compatibility information. In *Proceedings of ACL*.

# Subgroup Detection in Ideological Discussions

**Amjad Abu-Jbara**

EECS Department  
University of Michigan  
Ann Arbor, MI, USA  
amjbara@umich.edu

**Pradeep Dasigi**

Department of Computer Science  
Columbia University  
New York, NY, USA  
pd2359@columbia.edu

**Mona Diab**

Center for Computational Learning Systems  
Columbia University  
New York, NY, USA  
mdiab@ccls.columbia.edu

**Dragomir Radev**

EECS Department  
University of Michigan  
Ann Arbor, MI, USA  
radev@umich.edu

## Abstract

The rapid and continuous growth of social networking sites has led to the emergence of many communities of communicating groups. Many of these groups discuss ideological and political topics. It is not uncommon that the participants in such discussions split into two or more subgroups. The members of each subgroup share the same opinion toward the discussion topic and are more likely to agree with members of the same subgroup and disagree with members from opposing subgroups. In this paper, we propose an unsupervised approach for automatically detecting discussant subgroups in online communities. We analyze the text exchanged between the participants of a discussion to identify the attitude they carry toward each other and towards the various aspects of the discussion topic. We use attitude predictions to construct an attitude vector for each discussant. We use clustering techniques to cluster these vectors and, hence, determine the subgroup membership of each participant. We compare our methods to text clustering and other baselines, and show that our method achieves promising results.

## 1 Introduction

Online forums discussing ideological and political topics are common<sup>1</sup>. When people discuss a disputed topic they usually split into subgroups. The members of each subgroup carry the same opinion

<sup>1</sup>[www.politicalforum.com](http://www.politicalforum.com), [www.createdebate.com](http://www.createdebate.com), [www.forandagainst.com](http://www.forandagainst.com), etc

toward the discussion topic. The member of a subgroup is more likely to show positive attitude to the members of the same subgroup, and negative attitude to the members of opposing subgroups.

For example, let us consider the following two snippets from a debate about the enforcement of a new immigration law in Arizona state in the United States:

(1) *Discussant 1: Arizona immigration law is good. Illegal immigration is bad.*

(2) *Discussant 2: I totally disagree with you. Arizona immigration law is blatant racism, and quite unconstitutional.*

In (1), the writer is expressing positive attitude regarding the immigration law and negative attitude regarding illegal immigration. The writer of (2) is expressing negative attitude towards the writer of (1) and negative attitude regarding the immigration law. It is clear from this short dialog that the writer of (1) and the writer of (2) are members of two opposing subgroups. Discussant 1 is supporting the new law, while Discussant 2 is against it.

In this paper, we present an unsupervised approach for determining the subgroup membership of each participant in a discussion. We use linguistic techniques to identify attitude expressions, their polarities, and their targets. The target of attitude could be another discussant or an entity mentioned in the discussion. We use sentiment analysis techniques to identify opinion expressions. We use named en-

tivity recognition and noun phrase chunking to identify the entities mentioned in the discussion. The opinion-target pairs are identified using a number of syntactic and semantic rules.

For each participant in the discussion, we construct a vector of attitude features. We call this vector the *discussant attitude profile*. The attitude profile of a discussant contains an entry for every other discussant and an entry for every entity mentioned in the discussion. We use clustering techniques to cluster the attitude vector space. We use the clustering results to determine the subgroup structure of the discussion group and the subgroup membership of each participant.

The rest of this paper is organized as follows. Section 2 examines the previous work. We describe the data used in the paper in Section 2.4. Section 3 presents our approach. Experiments, results and analysis are presented in Section 4. We conclude in Section 5

## 2 Related Work

### 2.1 Sentiment Analysis

Our work is related to a huge body of work on sentiment analysis. Previous work has studied sentiment in text at different levels of granularity. The first level is identifying the polarity of individual words. Hatzivassiloglou and McKeown (1997) proposed a method to identify the polarity of adjectives based on conjunctions linking them. Turney and Littman (2003) used pointwise mutual information (PMI) and latent semantic analysis (LSA) to compute the association between a given word and a set of positive/negative seed words. Takamura et al. (2005) proposed using a spin model to predict word polarity. Other studies used WordNet to improve word polarity prediction (Hu and Liu, 2004a; Kamps et al., 2004; Kim and Hovy, 2004; Andreevskaia and Bergler, 2006). Hassan and Radev (2010) used a random walk model built on top of a word relatedness network to predict the semantic orientation of English words. Hassan et al. (2011) proposed a method to extend their random walk model to assist word polarity identification in other languages including Arabic and Hindi.

Other work focused on identifying the subjectivity of words. The goal of this work is to deter-

mine whether a given word is factual or subjective. We use previous work on subjectivity and polarity prediction to identify opinion words in discussions. Some of the work on this problem classifies words as factual or subjective regardless of their context (Wiebe, 2000; Hatzivassiloglou and Wiebe, 2000; Banea et al., 2008). Some other work noticed that the subjectivity of a given word depends on its context. Therefore, several studies proposed using contextual features to determine the subjectivity of a given word within its context (Riloff and Wiebe, 2003; Yu and Hatzivassiloglou, 2003; Nasukawa and Yi, 2003; Popescu and Etzioni, 2005).

The second level of granularity is the sentence level. Hassan et al. (2010) presents a method for identifying sentences that display an attitude from the text writer toward the text recipient. They define attitude as the mental position of one participant with regard to another participant. A very detailed survey that covers techniques and approaches in sentiment analysis and opinion mining could be found in (Pang and Lee, 2008).

### 2.2 Opinion Target Extraction

Several methods have been proposed to identify the target of an opinion expression. Most of the work have been done in the context of product reviews mining (Hu and Liu, 2004b; Kobayashi et al., 2007; Mei et al., 2007; Stoyanov and Cardie, 2008). In this context, opinion targets usually refer to product features (i.e. product components or attributes, as defined by Liu (2009)). In the work of Hu and Liu (2004b), they treat frequent nouns and noun phrases as product feature candidates. In our work, we extract as targets frequent noun phrases and named entities that are used by two or more different discussants. Scaffidi et al. (2007) propose a language model approach to product feature extraction. They assume that product features are mentioned more often in product reviews than they appear in general English text. However, such statistics may not be reliable when the corpus size is small.

In another related work, Jakob and Gurevych (2010) showed that resolving the anaphoric links in the text significantly improves opinion target extraction. In our work, we use anaphora resolution to improve opinion-target

Participant A posted:	I support Arizona because they have every right to do so. They are just upholding well-established federal law. All states should enact such a law.
Participant B commented on A's post:	I support the law because the federal government is either afraid or indifferent to the issue. Arizona has the right and the responsibility to protect the people of the State of Arizona. If this requires a possible slight inconvenience to any citizen so be it.
Participant C commented on B's post:	That is such a sad thing to say. You do realize that under the 14th Amendment, the very interaction of a police officer asking you to prove your citizenship is Unconstitutional? As soon as you start trading Constitutional rights for "security", then you've lost.

Table 1: Example posts from the Arizona Immigration Law thread

pairing as shown in Section 3 below.

### 2.3 Community Mining

Previous work also studied community mining in social media sites. Somasundaran and Wiebe (2009) presents an unsupervised opinion analysis method for debate-side classification. They mine the web to learn associations that are indicative of opinion stances in debates and combine this knowledge with discourse information. Anand et al. (2011) present a supervised method for stance classification. They use a number of linguistic and structural features such as unigrams, bigrams, cue words, repeated punctuation, and opinion dependencies to build a stance classification model. This work is limited to dual sided debates and defines the problem as a classification task where the two debate sides are known beforehand. Our work is characterized by handling multi-side debates and by regarding the problem as a clustering problem where the number of sides is not known by the algorithm. This work also utilizes only discussant-to-topic attitude predictions for debate-side classification. Our work utilizes both discussant-to-topic and discussant-to-discussant attitude predictions.

In another work, Kim and Hovy (2007) predict the results of an election by analyzing discussion threads in online forums that discuss the elections. They use a supervised approach that uses unigrams, bigrams, and trigrams as features. In contrast, our work is unsupervised and uses different types of information. Moreover, although this work is related to ours at the goal level, it does not involve any opinion analysis.

Another related work classifies the speakers side in a corpus of congressional floor debates, using the speakers final vote on the bill as a labeling for side (Thomas et al., 2006; Bansal et al., 2008;

Yessenalina et al., 2010). This work infers agreement between speakers based on cases where one speaker mentions another by name, and a simple algorithm for determining the polarity of the sentence in which the mention occurs. This work shows that even with the resulting sparsely connected agreement structure, the MinCut algorithm can improve over stance classification based on textual information alone. This work also requires that the debate sides be known by the algorithm and it only identifies discussant-to-discussant attitude. In our experiments below we show that identifying both discussant-to-discussant and discussant-to-topic attitudes achieves better results.

### 2.4 Data

In this section, we describe the datasets used in this paper. We use three different datasets. The first dataset (*politicalforum*, henceforth) consists of 5,743 posts collected from a political forum<sup>2</sup>. All the posts are in English. The posts cover 12 disputed political and ideological topics. The discussants of each topic were asked to participate in a poll. The poll asked them to determine their stance on the discussion topic by choosing one item from a list of possible arguments. The list of participants who voted for each argument was published with the poll results. Each poll was accompanied by a discussion thread. The people who participated in the poll were allowed to post text to that thread to justify their choices and to argue with other participants. We collected the votes and the discussion thread of each poll. We used the votes to identify the subgroup membership of each participant.

The second dataset (*createdebate*, henceforth) comes from an online debating site<sup>3</sup>. It consists of

<sup>2</sup><http://www.politicalforum.com>

<sup>3</sup><http://www.createdebate.com>

Source	Topic	Question	#Sides	#Posts	#Participants
Politicalforum	Arizona Immigration Law	Do you support Arizona in its decision to enact their Immigration Enforcement law?	2	738	59
	Airport Security	Should we pick muslims out of the line and give additional scrutiny/screening?	4	735	69
	Vote for Obama	Will you vote for Obama in the 2012 Presidential elections?	2	2599	197
Createdebate	Evolution	Has evolution been scientifically proved?	2	194	98
	Social networking sites	It is easier to maintain good relationships in social networking sites such as Facebook.	2	70	31
	Abortion	Should abortion be banned	3	477	70
Wikipedia	Ireland	Misleading description of Irland island partition	3	40	10
	South Africa Government	Was the current form of South African government born in May 1910?	3	23	5
	Oil Spill	Obama's response to gulf oil spill	3	30	12

Table 2: Example threads from our three datasets

30 debates containing a total of 2,712 posts. Each debate is about one topic. The description of each debate states two or more positions regarding the debate topic. When a new participant enters the discussion, she explicitly picks a position and posts text to support it, support a post written by another participant who took the same position, or to dispute a post written by another participant who took an opposing position. We collected the discussion thread and the participant positions for each debate.

The third dataset (*wikipedia*, henceforth) comes from the Wikipedia<sup>4</sup> discussion section. When a topic on Wikipedia is disputed, the editors of that topic start a discussion about it. We collected 117 Wikipedia discussion threads. The threads contains a total of 1,867 posts.

The *politicalforum* and *createdebate* datasets are self labeled as described above. To annotate the Wikipedia data, we asked an expert annotator (a professor in sociolinguistics who is not one of the authors) to read each of the Wikipedia discussion threads and determine whether the discussants split into subgroups in which case he was asked to determine the subgroup membership of each discussant.

Table 2 lists few example threads from our three datasets. Table 1 shows a portion of discussion thread between three participants about enforcing a new immigration law in Arizona. This thread appeared in the *politicalforum* dataset. The text posted by the three participants indicates that A's position

is with enforcing the law, that B agrees with A, and that C disagrees with both. This means that A and B belong to the same opinion subgroup, while belongs to an opposing subgroup.

We randomly selected 6 threads from our datasets (2 from *politicalforum*, 2 from *createdebate*, and 2 from *Wikipedia*) and used them as development set. This set was used to develop our approach.

### 3 Approach

In this section, we describe a system that takes a discussion thread as input and outputs the subgroup membership of each discussant. Figure 1 illustrates the processing steps performed by our system to detect subgroups. In the following subsections we describe the different stages in the system pipeline.

#### 3.1 Thread Parsing

We start by parsing the thread to identify posts, participants, and the reply structure of the thread (i.e. who replies to whom). In the datasets described in Section 2.4, all this information was explicitly available in the thread. We tokenize the text of each post and split it into sentences using CLAIRLib (Abu-Jbara and Radev, 2011).

#### 3.2 Opinion Word Identification

The next step is to identify the words that express opinion and determine their polarity (positive or negative). Lehrer (1974) defines word polarity as the direction the word deviates to from the norm. We

<sup>4</sup><http://www.wikipedia.com>

use OpinionFinder (Wilson et al., 2005a) to identify polarized words and their polarities.

The polarity of a word is usually affected by the context in which it appears. For example, the word *fine* is positive when used as an adjective and negative when used as a noun. For another example, a positive word that appears in a negated context becomes negative. OpinionFinder uses a large set of features to identify the contextual polarity of a given polarized word given its isolated polarity and the sentence in which it appears (Wilson et al., 2005b). Snippet (3) below shows the result of applying this step to snippet (1) above (*O* means neutral; *POS* means positive; *NEG* means negative).

(3) *Arizona/O Immigration/O law/O good/POS ./O Illegal/O immigration/O bad/NEG ./O*

### 3.3 Target Identification

The goal of this step is to identify the possible targets of opinion. A target could be another discussant or an entity mentioned in the discussion. When the target of opinion is another discussant, either the discussant name is mentioned explicitly or a second person pronoun is used to indicate that the opinion is targeting the recipient of the post. For example, in snippet (2) above the second person pronoun *you* indicates that the opinion word *disagree* is targeting *Discussant 1*, the recipient of the post.

The target of opinion can also be an entity mentioned in the discussion. We use two methods to identify such entities. The first method uses shallow parsing to identify noun groups (NG). We use the Edinburgh Language Technology Text Tokenization Toolkit (LT-TTT) (Grover et al., 2000) for this purpose. We consider as an entity any noun group that is mentioned by at least two different discussants. We replace each identified entity with a unique placeholder ( $ENTITY_{ID}$ ). For example, the noun group *Arizona immigration law* is mentioned by *Discussant 1* and *Discussant 2* in snippets 1 and 2 above respectively. Therefore, we replace it with a placeholder as illustrated in snippets (4) and (5) below.

(4) *Discussant 1: ENTITY<sub>1</sub> is good. Illegal im-*

NER	NP Chunking
Barack Obama	the Republican nominee
Middle East	the maverick economists
Bush	conservative ideologues
Bob McDonell	the Nobel Prize
Iraq	Federal Government

Table 3: Some of the entities identified using NER and NP Chunking in a discussion thread about the US 2012 elections

*migration is bad.*

(5) *Discussant 2: I totally disagree with you. ENTITY<sub>1</sub> is blatant racism, and quite unconstitutional.*

We only consider as entities noun groups that contain two words or more. We impose this requirement because individual nouns are very common and regarding all of them as entities will introduce significant noise.

In addition to this shallow parsing method, we also use named entity recognition (NER) to identify more entities. We use the Stanford Named Entity Recognizer (Finkel et al., 2005) for this purpose. It recognizes three types of entities: person, location, and organization. We impose no restrictions on the entities identified using this method. Again, we replace each distinct entity with a unique placeholder. The final set of entities identified in a thread is the union of the entities identified by the two aforementioned methods. Table 3

Finally, a challenge that always arises when performing text mining tasks at this level of granularity is that entities are usually expressed by anaphorical pronouns. Previous work has shown that for example, the following snippet contains an explicit mention of the entity *Obama* in the first sentence, and then uses a pronoun to refer to the same entity in the second sentence. The opinion word *unbeatable* appears in the second sentence and is syntactically related to the pronoun *He*. In the next subsection, it will become clear why knowing which entity does the pronoun *He* refers to is essential for opinion-target pairing.

(6) *It doesn't matter whether you vote for Obama.*

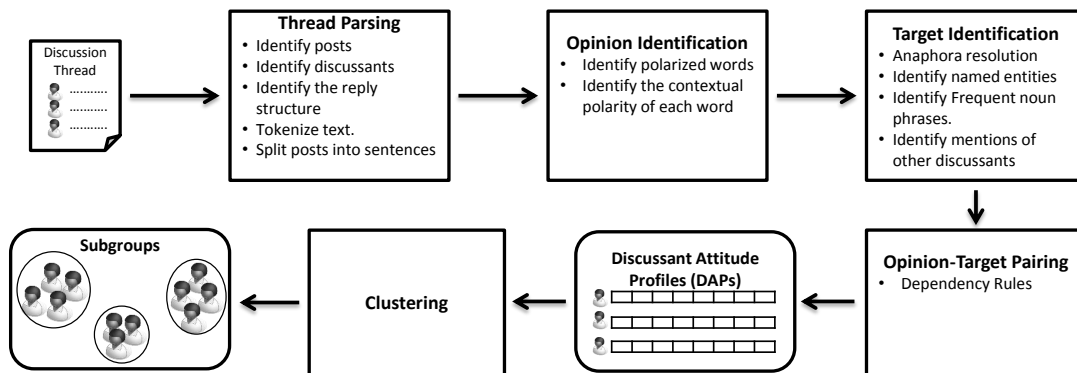


Figure 1: An overview of the subgroups detection system

*He is unbeatable.*

Jakob and Gurevych (2010) showed experimentally that resolving the anaphoric links in the text significantly improves opinion target extraction. We use the Beautiful Anaphora Resolution Toolkit (BART) (Versley et al., 2008) to resolve all the anaphoric links within the text of each post separately. The result of applying this step to snippet (6) is:

(6) *It doesn't matter whether you vote for Obama. Obama is unbeatable.*

Now, both mentions of *Obama* will be recognized by the Stanford NER system and will be identified as one entity.

### 3.4 Opinion-Target Pairing

At this point, we have all the opinion words and the potential targets identified separately. The next step is to determine which opinion word is targeting which target. We propose a rule based approach for opinion-target pairing. Our rules are based on the dependency relations that connect the words in a sentence. We use the Stanford Parser (Klein and Manning, 2003) to generate the dependency parse tree of each sentence in the thread. An opinion word and a target form a pair if they stratify at least one of our dependency rules. Table 4 illustrates some

of these rules <sup>5</sup>. The rules basically examine the types of the dependencies on the shortest path that connect the opinion word and the target in the dependency parse tree. It has been shown in previous work on relation extraction that the shortest dependency path between any two entities captures the information required to assert a relationship between them (Bunescu and Mooney, 2005).

If a sentence  $S$  in a post written by participant  $P_i$  contains an opinion word  $OP_j$  and a target  $TR_k$ , and if the opinion-target pair satisfies one of our dependency rules, we say that  $P_i$  expresses an attitude towards  $TR_k$ . The polarity of the attitude is determined by the polarity of  $OP_j$ . We represent this as  $P_i \xrightarrow{+} TR_k$  if  $OP_j$  is positive and  $P_i \xrightarrow{-} TR_k$  if  $OP_j$  is negative.

It is likely that the same participant  $P_i$  express sentiment toward the same target  $TR_k$  multiple times in different sentences in different posts. We keep track of the counts of all the instances of positive/negative attitude  $P_i$  expresses toward  $TR_k$ . We represent this as  $P_i \xrightarrow[m-]{m+} TR_k$  where  $m$  ( $n$ ) is the number of times  $P_i$  expressed positive (negative) attitude toward  $TR_k$ .

### 3.5 Discussant Attitude Profile

We propose a representation of discussants' attitudes towards the identified targets in the discussion thread. As stated above, a target could be another discussant or an entity mentioned in the discussion.

<sup>5</sup>The code will be made publicly available at the time of publication



ID	Rule	In Words	Example
R1	$OP \rightarrow nsubj \rightarrow TR$	The target $TR$ is the nominal subject of the opinion word $OP$	ENTITY1 $_{TR}$ is good $_{OP}$ .
R2	$OP \rightarrow dobj \rightarrow TR$	The target $T$ is a direct object of the opinion $OP$	I hate $_{OP}$ ENTITY2 $_{TR}$
R3	$OP \rightarrow prep_* \rightarrow TR$	The target $TR$ is the object of a preposition that modifies the opinion word $OP$	I totally disagree $_{OP}$ with you $_{TR}$ .
R4	$TR \rightarrow amod \rightarrow OP$	The opinion is an adjectival modifier of the target	The bad $_{OP}$ ENTITY3 $_{TR}$ is spreading lies
R5	$OP \rightarrow nsubjpass \rightarrow TR$	The target $TR$ is the nominal subject of the passive opinion word $OP$	ENTITY4 $_{TR}$ is hated $_{OP}$ by everybody.
R6	$OP \rightarrow prep_* \rightarrow poss \rightarrow TR$	The opinion word $OP$ connected through a $prep_*$ relation as in $R2$ to something possessed by the target $TR$	The main flaw $_{OP}$ in your $_{TR}$ analysis is that it's based on wrong assumptions.
R7	$OP \rightarrow dobj \rightarrow poss \rightarrow TR$	The target $TR$ possesses something that is the direct object of the opinion word $OP$	I like $_{OP}$ ENTITY5 $_{TR}$ 's brilliant ideas.
R8	$OP \rightarrow csubj \rightarrow nsubj \rightarrow TR$	The opinion word $OP$ is a causal subject of a phrase that has the target $TR$ as its nominal subject	What ENTITY6 $_{TR}$ announced was misleading $_{OP}$ .

Table 4: Examples of the dependency rules used for opinion-target pairing.

Our representation is a vector containing numerical values. The values correspond to the counts of positive/negative attitudes expressed by the discussant toward each of the targets. We call this vector the *discussant attitude profile (DAP)*. We construct a DAP for every discussant. Given a discussion thread with  $d$  discussants and  $e$  entity targets, each attitude profile vector has  $n = (d + e) * 3$  dimensions. In other words, each target (discussant or entity) has three corresponding values in the DAP: 1) the number of times the discussant expressed positive attitude toward the target, 2) the number of times the discussant expressed a negative attitude towards the target, and 3) the number of times the the discussant interacted with or mentioned the target. It has to be noted that these values are not symmetric since the discussions explicitly denote the source and the target of each post.

### 3.6 Clustering

At this point, we have an attitude profile (or vector) constructed for each discussant. Our goal is to use these attitude profiles to determine the subgroup membership of each discussant. We can achieve this goal by noticing that the attitude profiles of discussants who share the same opinion are more likely to be similar to each other than to the attitude profiles of discussants with opposing opinions. This suggests that clustering the attitude vector space will achieve the goal and split the discussants into subgroups according to their opinion.

## 4 Evaluation

In this section, we present several levels of evaluation of our system. First, we compare our system to baseline systems. Second, we study how the choice of the clustering algorithm impacts the results. Third, we study the impact of each component in our system on the performance. All the results reported in this section that show difference in the performance are statistically significant at the 0.05 level (as indicated by a 2-tailed paired t-test). Before describing the experiments and presenting the results, we first describe the evaluation metrics we use.

### 4.0.1 Evaluation Metrics

We use two evaluation metrics to evaluate subgroups detection accuracy: Purity and Entropy. To compute Purity (Manning et al., 2008), each cluster is assigned the class of the majority vote within the cluster, and then the accuracy of this assignment is measured by dividing the number of correctly assigned members by the total number of instances. It can be formally defined as:

$$purity(\Omega, C) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j| \quad (1)$$

where  $\Omega = \{\omega_1, \omega_2, \dots, \omega_k\}$  is the set of clusters and  $C = \{c_1, c_2, \dots, c_J\}$  is the set of classes.  $\omega_k$  is interpreted as the set of documents in  $\omega_k$  and  $c_j$  as

the set of documents in  $c_j$ . The purity increases as the quality of clustering improves.

The second metric is Entropy. The Entropy of a cluster reflects how the members of the  $k$  distinct subgroups are distributed within each resulting cluster; the global quality measure is computed by averaging the entropy of all clusters:

$$Entropy = - \sum_j \frac{n_j}{n} \sum_i P(i, j) \times \log_2 P(i, j) \quad (2)$$

where  $P(i, j)$  is the probability of finding an element from the category  $i$  in the cluster  $j$ ,  $n_j$  is the number of items in cluster  $j$ , and  $n$  the total number of items in the distribution. In contrast to purity, the entropy decreases as the quality of clustering improves.

#### 4.1 Comparison to Baseline Systems

We compare our system (DAPC) that was described in Section 3 to two baseline methods. The first baseline (GC) uses graph clustering to partition a network based on the interaction frequency between participants. We build a graph where each node represents a participant. Edges link participants if they exchange posts, and edge weights are based on the number of interactions. We tried two methods for clustering the resulting graph: spectral partitioning (Luxburg, 2007) and a hierarchical agglomeration algorithm which works by greedily optimizing the modularity for graphs (Clauset et al., 2004).

The second baseline (TC) is based on the premise that the member of the same subgroup are more likely to use vocabulary drawn from the same language model. We collect all the text posted by each participant and create a tf-idf representations of the text in a high dimensional vector space. We then cluster the vector space to identify subgroups. We use k-means (MacQueen, 1967) as our clustering algorithm in this experiment (comparison of various clustering algorithms is presented in the next subsection). The distances between vectors are Euclidean distances. Table 5 shows that our system performs significantly better the baselines on the three datasets in terms of both the purity ( $P$ ) and the entropy ( $E$ ) (notice that lower entropy values indicate better clustering). The values reported are the

Method	Createdebate		Politicalforum		Wikipedia	
	P	E	P	E	P	E
GC - Spectral	0.50	0.85	0.50	0.88	0.49	0.89
GC - Hierarchical	0.48	0.86	0.47	0.89	0.49	0.87
TC - kmeans	0.51	0.84	0.49	0.88	0.52	0.85
<b>DAPC - kmeans</b>	<b>0.64</b>	<b>0.68</b>	<b>0.61</b>	<b>0.80</b>	<b>0.66</b>	<b>0.55</b>

Table 5: Comparison to baseline systems

Method	Createdebate		Politicalforum		Wikipedia	
	P	E	P	E	P	E
DAPC - EM	0.63	0.71	0.61	0.82	0.63	0.61
DAPC - FF	0.63	0.70	0.60	0.83	0.64	0.59
DAPC - kmeans	0.64	0.68	0.61	0.80	0.66	0.55

Table 6: Comparison of different clustering algorithms

average results of the threads of each dataset. We believe that the baselines performed poorly because the interaction frequency and the text similarity are not key factors in identifying subgroup structures. Many people would respond to people they disagree with more, while others would mainly respond to people they agree with most of the time. Also, people in opposing subgroups tend to use very similar text when discussing the same topic and hence text clustering does not work as well.

#### 4.2 Choice of the clustering algorithm

We experimented with three different clustering algorithms: expectation maximization (EM), and k-means (MacQueen, 1967), and FarthestFirst (FF) (Hochbaum and Shmoys, 1985; Dasgupta, 2002). As we did in the previous subsection, we use Euclidean distance to measure the distance between vectors. All the system (DAP) components are included as described in Section 3. The purity and entropy values using each algorithm are shown in Table 6. Although k-means seems to be performing slightly better than other algorithms, the differences in the results are not significant. This indicates that the choice of the clustering algorithm does not have a noticeable impact on the results. We also experimented with using Manhattan distance and cosine similarity instead of Euclidean distance to measure the distance between attitude vectors. We noticed that the choice of the distance does not have significant impact on the results as well.

### 4.3 Component Evaluation

In this subsection, we evaluate the impact of the different components in the pipeline on the system performance. We do that by removing each component from the pipeline and measuring the change in performance. We perform the following experiments: 1) We run the full system with all its components included (DAPC). 2) We run the system and include only discussant-to-discussant attitude features in the attitude vectors (DAPC-DD). 3) We include only discussant-to-entity attitude features in the attitude vectors (DAPC-DE). 4) We include only sentiment features in the attitude vector; i.e. we exclude the interaction count features (DAPC-SE). 5) We include only interaction count features to the attitude vector; i.e. we exclude sentiment features (DAPC-INT). 6) We skip the anaphora resolution step in the entity identification component (DAPC-NO\_AR). 7) We only use named entity recognition to identify entity targets; i.e. we exclude the entities identified through noun phrasing chunking (DAPC-NER). 8) Finally, we only noun phrase chunking to identify entity targets (DAPC-NP). In all these experiments k-means is used for clustering and the number of clusters is set as explained in the previous subsection.

The results show that all the components in the system contribute to better performance of the system. We notice from the results that the performance of the system drops significantly if sentiment features are not included. This result corroborates our hypothesis that interaction features are not sufficient factors for detecting rift in discussion groups. Including interaction features improve the performance (although not by a big difference) because they help differentiate between the case where participants A and B never interacted with each other and the case where they interact several time but never posted text that indicate difference in opinion between them. We also notice that the performance drops significantly in DAPC-DD and DAPC-DE which also supports our hypotheses that both the sentiment discussants show toward one another and the sentiment they show toward the aspects of the discussed topic are important for the task. Although using both named entity recognition (NER) and noun phrase chunking achieves better results, it

Method	Createdebate		Politicalforum		Wikipedia	
	P	E	P	E	P	E
DAPC	<b>0.64</b>	<b>0.68</b>	<b>0.61</b>	<b>0.80</b>	<b>0.66</b>	<b>0.55</b>
DAPC-DD	0.59	0.77	0.57	0.86	0.62	0.61
DAPC-DE	0.60	0.69	0.58	0.84	0.58	0.78
DAPC-SE	0.62	0.70	0.60	0.83	0.61	0.62
DAPC-INT	0.54	0.88	0.52	0.91	0.57	0.85
DAPC-NO_AR	0.62	0.72	0.60	0.84	0.64	0.60
DAPC-NER	0.61	0.71	0.58	0.86	0.63	0.59
DAPC-NP	0.63	0.75	0.59	0.84	0.65	0.62

Table 7: Impact of system components on the performance

can also be noted from the results that NER contributes more to the system performance. Finally, the results support Jakob and Gurevych (2010) findings that anaphora resolution aids opinion mining systems.

## 5 Conclusions

In this paper, we presented an approach for subgroup detection in ideological discussions. Our system uses linguistic analysis techniques to identify the attitude the participants of online discussions carry toward each other and toward the aspects of the discussion topic. Attitude prediction as well as interaction frequency to construct an attitude vector for each participant. The attitude vectors of discussants are then clustered to form subgroups. Our experiments showed that our system outperforms text clustering and interaction graph clustering. We also studied the contribution of each component in our system to the overall performance.

## Acknowledgments

This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the U.S. Army Research Lab. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government.

## References

- Amjad Abu-Jbara and Dragomir Radev. 2011. Clairlib: A toolkit for natural language processing, information retrieval, and network analysis. In *Proceedings of the ACL-HLT 2011 System Demonstrations*, pages 121–126, Portland, Oregon, June. Association for Computational Linguistics.
- Pranav Anand, Marilyn Walker, Rob Abbott, Jean E. Fox Tree, Robeson Bowmani, and Michael Minor. 2011. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA 2.011)*, pages 1–9, Portland, Oregon, June. Association for Computational Linguistics.
- Alina Andreevskaia and Sabine Bergler. 2006. Mining wordnet for fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *EACL'06*.
- Carmen Banea, Rada Mihalcea, and Janyce Wiebe. 2008. A bootstrapping method for building subjectivity lexicons for languages with scarce resources. In *LREC'08*.
- Mohit Bansal, Claire Cardie, and Lillian Lee. 2008. The power of negative thinking: Exploiting label disagreement in the min-cut classification framework.
- Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 724–731, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.
- Aaron Clauset, Mark E. J. Newman, and Christopher Moore. 2004. Finding community structure in very large networks. *Phys. Rev. E*, 70:066111.
- Sanjoy Dasgupta. 2002. Performance guarantees for hierarchical clustering. In *15th Annual Conference on Computational Learning Theory*, pages 351–363. Springer.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Claire Grover, Colin Matheson, Andrei Mikheev, and Marc Moens. 2000. Lt ttt - a flexible tokenisation tool. In *In Proceedings of Second International Conference on Language Resources and Evaluation*, pages 1147–1154.
- Ahmed Hassan and Dragomir Radev. 2010. Identifying text polarity using random walks. In *ACL'10*.
- Ahmed Hassan, Vahed Qazvinian, and Dragomir Radev. 2010. What's with the attitude?: identifying sentences with attitude in online discussions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1245–1255.
- Ahmed Hassan, Amjad AbuJbara, Rahul Jha, and Dragomir Radev. 2011. Identifying the semantic orientation of foreign words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 592–597, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *EACL'97*, pages 174–181.
- Vasileios Hatzivassiloglou and Janyce Wiebe. 2000. Effects of adjective orientation and gradability on sentence subjectivity. In *COLING*, pages 299–305.
- Hochbaum and Shmoys. 1985. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184.
- Minqing Hu and Bing Liu. 2004a. Mining and summarizing customer reviews. In *KDD'04*, pages 168–177.
- Minqing Hu and Bing Liu. 2004b. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.
- Niklas Jakob and Iryna Gurevych. 2010. Using anaphora resolution to improve opinion target identification in movie reviews. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 263–268, Uppsala, Sweden, July. Association for Computational Linguistics.
- Jaap Kamps, Maarten Marx, Robert J. Mokken, and Maarten De Rijke. 2004. Using wordnet to measure semantic orientations of adjectives. In *National Institute for*, pages 1115–1118.
- Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *COLING*, pages 1367–1373.
- Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *IN PROCEEDINGS OF THE 41ST ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, pages 423–430.
- Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.
- Adrienne Lehrer. 1974. Semantic fields and lexical structure. North Holland, Amsterdam and New York.

- Bing Liu. 2009. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications)*. Springer, 1st ed. 2007. corr. 2nd printing edition, January.
- Ulrike Luxburg. 2007. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, December.
- J. B. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.
- Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 171–180, New York, NY, USA. ACM.
- Soo min Kim and Eduard Hovy. 2007. Crystal: Analyzing predictive opinions on the web. In *In EMNLP-CoNLL 2007*.
- Tetsuya Nasukawa and Jeonghee Yi. 2003. Sentiment analysis: capturing favorability using natural language processing. In *K-CAP '03: Proceedings of the 2nd international conference on Knowledge capture*, pages 70–77.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.
- Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *HLT-EMNLP'05*, pages 339–346.
- Ellen Riloff and Janyce Wiebe. 2003. Learning extraction patterns for subjective expressions. In *EMNLP'03*, pages 105–112.
- Swapna Somasundaran and Janyce Wiebe. 2009. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 226–234, Suntec, Singapore, August. Association for Computational Linguistics.
- Veselin Stoyanov and Claire Cardie. 2008. Topic identification for fine-grained opinion analysis. In *In Coling*.
- Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientations of words using spin model. In *ACL'05*, pages 133–140.
- Matt Thomas, Bo Pang, and Lillian Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *In Proceedings of EMNLP*, pages 327–335.
- Peter Turney and Michael Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21:315–346.
- Yannick Versley, Simone Paolo Ponzetto, Massimo Poesio, Vladimir Eidelman, Alan Jern, Jason Smith, Xiaofeng Yang, and Alessandro Moschitti. 2008. Bart: A modular toolkit for coreference resolution. In *Proceedings of the ACL-08: HLT Demo Session*, pages 9–12, Columbus, Ohio, June. Association for Computational Linguistics.
- Janyce Wiebe. 2000. Learning subjective adjectives from corpora. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 735–740.
- Theresa Wilson, Paul Hoffmann, Swapna Somasundaran, Jason Kessler, Janyce Wiebe, Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005a. Opinionfinder: a system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, HLT-Demo '05, pages 34–35, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005b. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/EMNLP'05*, Vancouver, Canada.
- Ainur Yessenalina, Yisong Yue, and Claire Cardie. 2010. Multi-level structured models for document-level sentiment classification. In *In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences. In *EMNLP'03*, pages 129–136.

# Cross-Domain Co-Extraction of Sentiment and Topic Lexicons

Fangtao Li<sup>§</sup>, Sinno Jialin Pan<sup>†</sup>, Ou Jin<sup>‡</sup>, Qiang Yang<sup>‡</sup> and Xiaoyan Zhu<sup>§</sup>

<sup>§</sup>Department of Computer Science and Technology, Tsinghua University, Beijing, China

<sup>§</sup>{fangtao06@gmail.com, zxy-dcs@tsinghua.edu.cn}

<sup>†</sup>Institute for Infocomm Research, Singapore

<sup>†</sup>jspan@i2r.a-star.edu.sg

<sup>‡</sup>Hong Kong University of Science and Technology, Hong Kong, China

<sup>‡</sup>{kingomiga@gmail.com, qyang@cse.ust.hk}

## Abstract

Extracting sentiment and topic lexicons is important for opinion mining. Previous works have showed that supervised learning methods are superior for this task. However, the performance of supervised methods highly relies on manually labeled training data. In this paper, we propose a domain adaptation framework for sentiment- and topic- lexicon co-extraction in a domain of interest where we do not require any labeled data, but have lots of labeled data in another related domain. The framework is twofold. In the first step, we generate a few high-confidence sentiment and topic seeds in the target domain. In the second step, we propose a novel Relational Adaptive bootstraPping (RAP) algorithm to expand the seeds in the target domain by exploiting the labeled source domain data and the relationships between topic and sentiment words. Experimental results show that our domain adaptation framework can extract precise lexicons in the target domain without any annotation.

## 1 Introduction

In the past few years, opinion mining and sentiment analysis have attracted much attention in Natural Language Processing (NLP) and Information Retrieval (IR) (Pang and Lee, 2008; Liu, 2010). Sentiment lexicon construction and topic lexicon extraction are two fundamental subtasks for opinion mining (Qiu et al., 2009). A sentiment lexicon is a list of sentiment expressions, which are used to indicate sentiment polarity (e.g., positive or negative). The sentiment lexicon is domain dependent as users may use different sentiment words to express their opinion in different domains (e.g., different products). A topic lexicon is a list of topic expressions, on which

the sentiment words are expressed. Extracting the topic lexicon from a specific domain is important because users not only care about the overall sentiment polarity of a review but also care about which aspects are mentioned in review. Note that, similar to sentiment lexicons, different domains may have very different topic lexicons.

Recently, Jin and Ho (2009) and Li *et al.* (2010a) showed that supervised learning methods can achieve state-of-the-art results for lexicon extraction. However, the performance of these methods highly relies on manually annotated training data. In most cases, the labeling work may be time-consuming and expensive. It is impossible to annotate each domain of interest to build precise domain-dependent lexicons. It is more desirable to automatically construct precise lexicons in domains of interest by transferring knowledge from other domains.

In this paper, we focus on the co-extraction task of sentiment and topic lexicons in a target domain where we do not have any labeled data, but have plenty of labeled data in a source domain. Our goal is to leverage the knowledge extracted from the source domain to help lexicon co-extraction in the target domain. To address this problem, we propose a two-stage domain adaptation method. In the first step, we build a bridge between the source and target domains by identifying some *common* sentiment words as sentiment seeds in the target domain, such as “good”, “bad”, “nice”, etc. After that, we generate topic seeds in the target domain by mining some *general* syntactic relation patterns between the sentiment and topic words from the source domain. In the second step, we propose a Relational Adaptive bootstraPping (RAP) algorithm to expand the seeds in the target domain. Our proposed method can uti-

lize useful labeled data from the source domain as well as exploit the relationships between the topic and sentiment words to propagate information for lexicon construction in the target domain. Experimental results show that our proposed method is effective for cross-domain lexicon co-extraction.

In summary, we have three main contributions: 1) We give a systematic study on cross-domain sentiment analysis in word level. While, most of previous work focused on document level; 2) A new two-step domain adaptation framework, with a novel RAP algorithm for seed expansion, is proposed. 3) We conduct extensive evaluation, and the experimental results demonstrate the effectiveness of our methods.

## 2 Related Work

### 2.1 Sentiment or Topic Lexicon Extraction

Sentiment or topic lexicon extraction is to identify the sentiment or topic words from text. In the past, many machine learning techniques have been proposed for this task. Hu and Liu *et al.* (2004) proposed an association-rule-based method to extract topic words and a dictionary-based method to identify sentiment words, independently. Wiebe *et al.* (2004) and Rioff *et al.* (2003) proposed to identify subjective adjectives and nouns using word clustering based on their distributional similarity. Popescu and Etzioni (2005) proposed a relaxed labeling approach to utilize linguistic rules for opinion polarity detection. Some researchers also proposed to use topic modeling to identify implicit topics and sentiment words (Mei *et al.*, 2007; Titov and McDonald, 2008; Zhao *et al.*, 2010; Li *et al.*, 2010b), where a topic is a cluster of words, which is different from our fine-grained topic-word extraction.

Jin and Ho (2009) and Li *et al.* (2010a) both proposed to use supervised sequential labeling methods for topic and opinion extraction. Experimental results showed that the supervised learning methods can achieve state-of-the-art performance on lexicon extraction. However, these methods need to manually annotate a lot of training data in each domain. Recently, Qiu *et al.* (2009) proposed a rule-based semi-supervised learning methods for lexicon extraction. However, their method requires to manually define some *general* syntactic rules among sentiment and topic words. In addition, it still requires some annotated words in the target domain. In this paper, we do not assume any predefined rules and

labeled data be available in the target domain.

### 2.2 Domain Adaptation

Domain adaptation aims at transferring knowledge across domains where data distributions may be different (Pan and Yang, 2010). In the past few years, domain adaptation techniques have been widely applied to various NLP tasks, such as part-of-speech tagging (Ando and Zhang, 2005; Jiang and Zhai, 2007; Daumé III, 2007), named-entity recognition and shallow parsing (Daumé III, 2007; Jiang and Zhai, 2007; Wu *et al.*, 2009). There are also lots of studies for cross-domain sentiment analysis (Blitzer *et al.*, 2007; Tan *et al.*, 2007; Li *et al.*, 2009; Pan *et al.*, 2010; Bollegala *et al.*, 2011; He *et al.*, 2011; Glorot *et al.*, 2011). However, most of them focused on coarse-grained document-level sentiment classification, which is different from our fine-grained word-level extraction. Our work is similar to Jakob and Gurevych (2010) which proposed a Conditional Random Field (CRF) for cross-domain topic word extraction. However, the performance of their method highly depends on the manually designed features. In our experiments, we compare our method with theirs, and find that ours can achieve much better results on cross-domain lexicon extraction. Note that our work is also different from a recent work (Du *et al.*, 2010), which focused on identifying the polarity of adjective words by using cross-domain knowledge. While we extract both topic and sentiment words and allow non-adjective sentiment words, which is more practical.

## 3 Cross-Domain Lexicon Co-Extraction

### 3.1 Problem Definition

Recall that, we focus on the setting where we have no labeled data in the target domain, while we have plenty of labeled data in the source domain. Denote  $\mathcal{D}_S = \{(w_{S_i}, y_{S_i})\}_{i=1}^{n_1}$  the source domain data, where  $w_{S_i}$  represents a word in the source domain.  $y_{S_i} \in \mathcal{Y}$  is the corresponding label of  $w_{S_i}$ . Similarly, we denote  $\mathcal{D}_T = \{w_{T_j}\}_{j=1}^{n_2}$  the target domain data, where the input  $w_{T_j}$  is a word in the target domain. In lexicon extraction,  $\mathcal{Y} \in \{1, 2, 3\}$ , where  $y_i = 1$  denotes the corresponding word  $w_i$  a sentiment word,  $y_i = 2$  denotes  $w_i$  a topic word, and  $y_i = 3$  denotes  $w_i$  neither a sentiment nor topic word. Our goal is to predict labels on  $\mathcal{D}_T$  to extract topic and sentiment words for constructing topic and

sentiment lexicons, respectively.

### 3.2 Motivating Examples

In this section, we use some examples to introduce the motivation behind our proposed method. Table 1 shows several reviews from two domains: *movie* and *camera*. From the table, we can observe that there are some common sentiment words across different domains, such as “great”, “excellent” and “amazing”. However, the topic words may be different. For example, in the movie domain, topic words include “movie” and “script”. While in the camera domain, topic words include “camera” and “photos”.

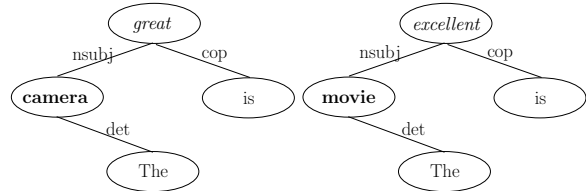
Domain	Review
camera	The <b>camera</b> is <i>great</i> .
	it is a very <i>amazing</i> <b>product</b> .
	i highly <i>recommend</i> this <b>camera</b> .
	takes <i>excellent</i> <b>photos</b> .
	<b>photos</b> had some <i>artifacts</i> and <i>noise</i> .
movie	This <b>movie</b> has <i>good</i> <b>script</b> , <i>great</i> <b>casting</b> , <i>excellent</i> <b>acting</b> .
	I <i>love</i> this <b>movie</b> .
	<b>Godfather</b> was the most <i>amazing</i> <b>movie</b> .
	The <b>movie</b> is <i>excellent</i> .

Table 1: Reviews in *camera* and *movie* domains. Boldfaces are topic words and Italics are sentiment words.

Based on the observations, we can build a connection between the source and target domains by identifying the common sentiment words. Furthermore, intuitively, there are some general syntactic relationships or patterns between topic and sentiment words across different domains. Therefore, if we can mine the patterns from the source and target domain data, then we are able to construct an indirect connection between topic words across domains by using the common sentiment words as a bridge, which makes knowledge transfer across domains possible.

Figure 1 shows two dependency trees for the sentence “the camera is great” in the camera domain and the sentence “the movie is excellent” in the movie domain, respectively. As can be observed, the relationships between the topic and sentiment words in the two sentences are the same. They both share a “TOPIC-nsubj-SENTIMENT” relation. Let the camera domain be the source domain and the movie domain be the target domain. If the word “excellent” is identified as a common sentiment word, and the “TOPIC-nsubj-SENTIMENT” relation extracted from the camera domain is recognized as a common

syntactic pattern, then the word “movie” can be predicted as a topic word in the movie domain with high probability. After new topic words are extracted in the movie domain, we can apply the same syntactic pattern or other syntactic patterns to extract new sentiment and topic words iteratively.



(a) Camera domain.

(b) Movie domain.

Figure 1: Examples of dependency tree structure.

More specifically, we use the shortest path between a topic word and a sentiment word in the corresponding dependency tree to denote the relation between them. To get more general paths, we do not take original words in the path into consideration, but use their POS tags instead, such as “NN”, “VB”, “JJ”, etc. As an example shown in Figure 2, we can extract two paths or relationships between topic and sentiment words from the dependency tree of the sentence “The movie has good script”: “NN-amod-JJ” from “script” and “good”, and “NN-nsubj-VB-dobj-NN-amod-JJ” from “movie” and “good”.

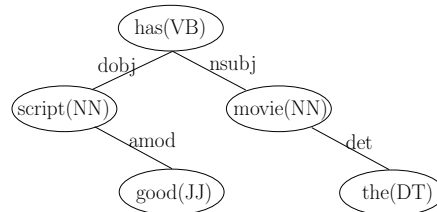


Figure 2: Example of pattern extraction.

In the following sections, we present the proposed two-stage domain adaptation framework: 1) generating some sentiment and topic seeds in the target domain; and 2) expanding the seeds in the target domain to construct sentiment and topic lexicons.

## 4 Seed Generation

Our basic idea is to first identify several *common* sentiment words across domains as sentiment seeds. Meanwhile, we mine some general patterns between sentiment and topic words from the source domain. Finally, we use the sentiment seeds and general patterns to generate topic seeds in the target domain.



#### 4.1 Sentiment Seed Generation

To identify *common* sentiment words across domains, we extract all sentiment words from the source domain as candidates. For each candidate, we calculate its score based on the following metric:

$$S_1(w_i) = (p_S(w_i) + p_T(w_i)) e^{(-|p_S(w_i) - p_T(w_i)|)}, \quad (1)$$

where  $p_S(w_i)$  and  $p_T(w_i)$  are the probabilities of the word  $w_i$  occurring in the source and target domains, respectively. If a word  $w_i$  has high  $S_1$  score, which implies that the word  $w_i$  occurs frequently and similarly in both domains, then it can be considered as a *common* sentiment word (Pan et al., 2010; Blitzer et al., 2007). We select top  $r$  candidates with highest  $S_1$  scores as sentiment seeds.

#### 4.2 Topic Seed Generation

We extract all patterns between sentiment and topic words in the source domain as candidates. For each pattern candidate, we calculate its score based on a metric defined in AutoSlog-TS (Riloff, 1996):

$$S_2(R_j) = Acc(R_j) \times \log_2(Freq(R_j)), \quad (2)$$

where  $Acc(R_j)$  is the accuracy of the pattern  $R_j$  in the source domain, and  $Freq(R_j)$  is the frequency of the pattern  $R_j$  observed in target domain. This metric aims to identify the patterns that are precise in the source domain and observed frequently in the target domain. We also select the top  $r$  patterns with highest  $S_2$  scores. With the patterns and sentiment seeds, we extract topic-word candidates and measure their scores based on a variant metric of quadratic combination (Zhang and Ye, 2008):

$$S_3(w_k) = \sum_{R_j \in A, w_i \in B} (S_2(R_j) \times S_1(w_i)), \quad (3)$$

where  $B$  is a set of sentiment seeds and  $A$  is a set of patterns which the words  $w_i$  and  $w_k$  satisfy. We then select the top  $r$  candidates as topic seeds.

### 5 Seed Expansion

After generating the topic and sentiment seeds, we aim to expand them in the target domain to construct topic and sentiment lexicons. In this section, we propose a new bootstrapping-based method to address this problem.

Bootstrapping is the process of improving the performance of a weak classifier by iteratively adding training data and retraining the classifier. More specifically, bootstrapping starts with a small set of labeled “seeds”, and iteratively adds unlabeled

data that are labeled by the classifier to the training set based on some selection criterion, and retrain the classifier. Many bootstrapping-based algorithms have been proposed to information extraction and other NLP tasks (Blum and Mitchell, 1998; Riloff and Jones, 1999; Jones et al., 1999; Wu et al., 2009).

One important issue in bootstrapping is how to design a criterion to select unlabeled data to be added to the training set iteratively. Our proposed bootstrapping for cross-domain lexicon extraction is based on the following two observations: 1) Although the source and target domains are different, part of source domain labeled data is still useful for lexicon extraction in the target domain after some adaptation; 2) The syntactic relationships among sentiment and topic words can be used to expand the seeds in the target domain for lexicon construction.

Based on the two observations, we propose a new bootstrapping-based method named Relational Adaptive bootstrapping (RAP), as summarized in Algorithm 1, for expanding lexicons across domains. In each iteration, we employ a cross-domain classifier trained on the source domain lexicons and the extracted target domain lexicons to predict the labels of the target unlabeled data, and select top  $k_2$  predicted topic and sentiment words as candidates based on confidence. With the extracted syntactic patterns in the previous iterations, we construct a bipartite graph between sentiment and topic words on the extracted target domain lexicons and candidates. After that, a graph-based score refinement algorithm is performed on the graph, and the top  $k_1$  candidates are added to the extracted lexicons based on the final scores. Accordingly, with the new extracted lexicons, we update the syntactic patterns in each iteration. The details of RAP are presented in the following sections.

#### 5.1 Cross-Domain Classifier

In this paper, we employ *Transfer AdaBoost* (TrAdaBoost) (Dai et al., 2007) as the cross-domain learning algorithm in RAP. In TrAdaBoost, each word  $w_{S_i}$  (or  $w_{T_j}$ ) is represented by a feature vector  $x_{S_i}$  (or  $x_{T_j}$ ). A classifier trained on the source domain data  $\mathcal{D}_S = \{(x_{S_i}, y_{S_i})\}$  may perform poor on  $x_{T_j}$  because of domain difference. The main idea of TrAdaBoost is to re-weight the source domain data based on a few of target domain labeled data, which is referred to as seeds in our task. The re-weighting

aims to reduce the effect of the “bad” source domain data while encourage the “good” ones to get a more precise classifier in target domain. In each iteration of RAP, we train cross-domain classifiers  $f_{\mathcal{O}}^T$  and  $f_{\mathcal{P}}^T$  for sentiment- and topic- word extraction using TrAdaBoost separately (taking sentiment or topic words as positive instances). We use linear Support Vector Machines (SVMs) as the base classifier in TrAdaBoost. For features to represent each word, we use lexicon features, such as the previous, current and next words, and POS tag features, such as the previous, current and next words’ POS tags.

### Algorithm 1 Relational Adaptive bootstrapping

**Require:** Target domain data  $\mathcal{D}_T = \mathcal{D}_T^l \cup \mathcal{D}_T^u$ , where  $\mathcal{D}_T^l$  consists of sentiment seeds  $B$  and topic seeds  $C$  and their initial scores  $S_1(w_i), \forall w_i \in B$  and  $S_3(w_j), \forall w_j \in C$ ,  $\mathcal{D}_T^u$  is the set of unlabeled target domain data; labeled source domain data  $\mathcal{D}_S$ ; a cross-domain classifier; iteration number  $M$  and candidate selection number  $k_1, k_2$ .

**Ensure:** Expand  $C$  and  $B$  in the target domain.

- 1: Initialize a pattern set  $A = \emptyset$ ,  $\tilde{S}_1(w_i) = S_1(w_i)$ ,  $w_i \in B$  and  $\tilde{S}_3(w_j) = S_3(w_j)$ ,  $w_j \in C$ . Consider all patterns observed in the source domain as pattern candidates  $P$ .
- 2: **for**  $m = 1 \dots M$  **do**
- 3: Extract new pattern candidates to  $P$  with  $\mathcal{D}_T^l$  in target domain, update pattern score  $\tilde{S}_2(R_j)$ , where  $R_j \in P$ , based on Eq. (4), and select the top  $k_1$  patterns to the pattern set  $A$ .
- 4: Learn the cross-domain classifiers  $f_{\mathcal{O}}^T$  and  $f_{\mathcal{P}}^T$  for sentiment- and topic- word extraction with  $\mathcal{D}_S \cup \mathcal{D}_T^l$  separately. Predict the sentiment score  $h_{f_{\mathcal{O}}^T}^T(w_{T_j})$  and topic score  $h_{f_{\mathcal{P}}^T}^T(w_{T_j})$  on  $\mathcal{D}_T^u$ , and select  $k_2$  sentiment words and topic words with highest scores as candidates.
- 5: Construct a bipartite graph between sentiment and topic words on  $\mathcal{D}_T^l$  and the  $k_2$  sentiment- and topic- word candidates, and calculate the normalized weights  $\theta_{ij}$ ’s for each edge of the graph.
- 6: Refine the scores  $\tilde{S}_1$  and  $\tilde{S}_3$  of the  $k_2$  sentiment and topic word candidates using Eqs. (5) and (6) iteratively.
- 7: Select  $k_1$  new sentiment words and  $k_1$  new topic words with the final scores, and add them to lexicons  $B$  and  $C$ . Update  $\tilde{S}_1(w_i)$  and  $\tilde{S}_3(w_j)$  accordingly.
- 8: **end for**
- 9: **return** Expanded lexicons  $B$  and  $C$ .

## 5.2 Graph Construction

Based on the cross-domain classifiers  $f_{\mathcal{O}}^T$  and  $f_{\mathcal{P}}^T$ , we can predict the sentiment label score  $h_{f_{\mathcal{O}}^T}^T(w_{T_i})$  and topic label score  $h_{f_{\mathcal{P}}^T}^T(w_{T_i})$  for the target domain data  $w_{T_i}$ . According to all predicted values, we respectively select top  $k_2$  new sentiment- and topic- words as candidates. Together with the extracted sentiment and topic lexicons in the target domain,

we build a bipartite graph among them as shown in Figure 3. In the bipartite graph, one set of nodes represents topic words, including new topic candidates and words in the lexicon  $C$ , and the other set of nodes represents sentiment words, including new sentiment candidates and words in the lexicon  $B$ . For a pair of sentiment and topic words  $w_{T_i}^{\mathcal{O}}$  and  $w_{T_j}^{\mathcal{P}}$ , if there is a pattern  $R_j$  in the pattern set  $A$  that they can satisfy, then there exists an edge  $e_{ij}$  between them. Furthermore, each edge  $e_{ij}$  is associated with a nonnegative weight  $\theta_{ij}$ , which is measured as follows,  $\theta_{ij} = \sum_{R_k \in E} \tilde{S}_2(R_k)$ , where  $\tilde{S}_2$  is the pattern score. Similar to the metric defined in Eq. (3), the pattern score is defined as:

$$\tilde{S}_2(R_j) = \sum_{\{w_i, w_k\} \in E} (\tilde{S}_1(w_i) \times \tilde{S}_3(w_k)), \quad (4)$$

where  $E = \{\{w_i, w_j\} | w_i \in B, w_j \in C \text{ and } w_i, w_j \text{ satisfy } R_j, R_j \in A\}$ . Note that in the beginning of each iteration,  $\tilde{S}_2$  is updated based on the new sentiment score  $\tilde{S}_1$  and topic score  $\tilde{S}_3$ . We further normalize  $\theta_{ij}$  by  $\hat{\theta}_{ij} = \theta_{ij} / (\sum_{ij} \theta_{ij})$ .

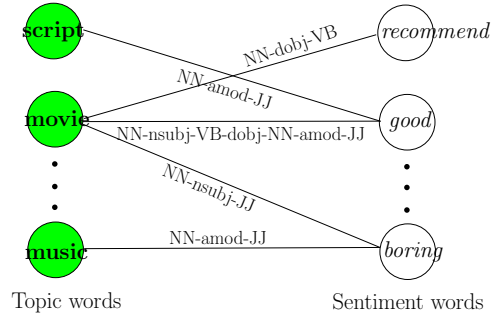


Figure 3: Topic and sentiment word graph.

## 5.3 Score Computation

We construct the bipartite graph to exploit the relationships between sentiment and topic words to propagate information for lexicon extraction. We use the following reinforcement formulas to iteratively update the final sentiment score  $\tilde{S}_1(w_{T_j})$  and topic score  $\tilde{S}_3(w_{T_i})$ , respectively:

$$\tilde{S}_1(w_{T_j}) = \mu \sum_i \tilde{S}_3(w_{T_i}) \hat{\theta}_{ij} + (1 - \mu) h_{f_{\mathcal{O}}^T}^T(w_{T_j}), \quad (5)$$

$$\tilde{S}_3(w_{T_i}) = \mu \sum_j \tilde{S}_1(w_{T_j}) \hat{\theta}_{ij} + (1 - \mu) h_{f_{\mathcal{P}}^T}^T(w_{T_i}), \quad (6)$$

where  $\mu$  is a trade-off parameter between the predicted value by cross-domain classifier and the reinforcement scores from other nodes connected by

edge  $e_{ij}$ . Here  $\mu$  is empirically set to be 0.5. With Eqs. (5) and (6), the sentiment scores and topic scores are iteratively refined until the state of the graph trends to be stable. This can be considered as an extension to the HITS algorithm (Kleinberg, 1999). Finally, we select  $k_1 \ll k_2$  sentiment and topic words from the  $k_2$  candidates based on their refined scores, and add them to the target domain lexicons, respectively. We also update the sentiment score  $\tilde{S}_1$  and topic score  $\tilde{S}_3$  for next iteration.

#### 5.4 Special Cases

We now introduce two special cases of the RAP algorithm. In Eqs. (5) and (6), if the parameter  $\mu = 1$ , then RAP only uses the relationships between sentiment and topic words with their patterns to propagate label information in the target domain without using the cross-domain classifier. We call this reduction relational bootstrapping. If  $\mu = 0$ , then RAP only utilizes useful source domain labeled data to assist learning of the target domain classifier without considering the relationships between sentiment and topic words. We call this reduction adaptive bootstrapping, which can be considered as a bootstrapping version of TrAdaBoost. We also empirically study these two special cases in experiments.

## 6 Experiments on Lexicon Evaluation

### 6.1 Data Set and Evaluation Criteria

We use the review dataset from (Li et al., 2010a), which contains 500 movie and 601 product reviews, for evaluation. The sentiment and topic words are manually annotated. In this dataset, all types of sentiment words are annotated instead of adjective words only. For example, the verbs, such as “like”, “recommend”, and nouns, such as “masterpiece”, are also labeled as sentiment words. We construct two cross-domain lexicon extraction tasks: “product vs. movie” and “movie vs. product”, where the word before “vs.” corresponds with the source domain and the word after “vs.” corresponds with the target domain. We evaluate our methods in terms of precision, recall and F-score ( $F1$ ).

### 6.2 Baselines

The results of in-domain classifiers, which are trained on plenty of target domain labeled data, can be treated as upper-bounds. We denote iSVM and iCRF the in-domain SVM and CRF classifiers in experiments, and compare our proposed methods,

RAP, relational bootstrapping, and adaptive bootstrapping, with the following baselines,

**Unsupervised Method (Un)** we implement a rule-based method for lexicon extraction based on (Hu and Liu, 2004), where adjective words that match a rule is recognized as sentiment words, and nouns that match a rule are recognized as topic words.

**Semi-Supervised Method (Semi)** we implement the double propagation model proposed in (Qiu et al., 2009). Since this method requires some target domain labeled data, we manually label 30 sentiment words in the target domain.

**Cross-Domain CRF (Cross-CRF)** we implement a cross-domain CRF algorithm proposed by (Jakob and Gurevych, 2010).

**TrAdaBoost** We apply TrAdaBoost (Dai et al., 2007) on the source domain labeled data and the generated seeds in the target domain to train a lexicon extractor.

### 6.3 Comparison Results

Comparison results on lexicon extraction are shown in Table 2 and Table 3. From Table 2, we can observe that our proposed methods are effective for sentiment lexicon extraction. The relational bootstrapping method performs better than the unsupervised method, TrAdaBoost and the cross-domain CRF algorithm, and achieves comparable results with the semi-supervised method. However, compared to the semi-supervised method, our proposed relational bootstrapping method does not require any labeled data in the target domain. We can also observe that the adaptive bootstrapping method and the RAP method perform much better than other methods in terms of F-score. The reason is that part of the source domain labeled data may be useful for learning the target classifier after reweighting. In addition, we also observe that embedding the TrAdaBoost algorithm into a bootstrapping process can further boost the performance of the classifier for sentiment lexicon extraction.

Table 3 shows the comparison results on topic lexicon extraction. From the table, we can observe that different from the sentiment lexicon extraction task, the relational bootstrapping method performs better than the adaptive bootstrapping method slightly. The reason may be that for the sentiment lexicon extraction task, there exist some common sentiment words

	<i>product vs. movie</i>			<i>movie vs. product</i>		
	Prec.	Rec.	<i>F1</i>	Prec.	Rec.	<i>F1</i>
Un	0.82	0.31	0.45	0.74	0.23	0.35
Semi	0.71	0.44	0.54	0.62	0.45	0.52
Cross-CRF	0.69	0.40	0.51	0.65	0.34	0.45
Tradaboost	0.73	0.41	0.52	0.72	0.42	0.52
Adaptive	0.68	0.53	0.59	0.63	0.52	0.57
Relational	0.55	0.51	0.53	0.57	0.51	0.54
RAP	0.69	0.59	<b>0.64</b>	0.66	0.59	<b>0.62</b>
iSVM	0.82	0.60	0.70	0.80	0.61	0.68
iCRF	0.80	0.66	0.72	0.80	0.62	0.69

Table 2: Results on sentiment lexicon extraction. Numbers in boldface denote significant improvement.

	<i>product vs. movie</i>			<i>movie vs. product</i>		
	Prec.	Rec.	<i>F1</i>	Prec.	Rec.	<i>F1</i>
Un	0.41	0.32	0.36	0.53	0.35	0.41
Semi	0.54	0.59	0.56	0.75	0.50	0.60
Cross-CRF	0.70	0.23	0.34	0.80	0.24	0.37
Tradaboost	0.64	0.45	0.53	0.57	0.47	0.51
Adaptive	0.76	0.44	0.56	0.70	0.52	0.59
Relational	0.57	0.58	0.58	0.61	0.57	0.59
RAP	0.80	0.56	<b>0.66</b>	0.73	0.58	<b>0.65</b>
iSVM	0.83	0.73	0.78	0.85	0.70	0.77
iCRF	0.84	0.78	0.81	0.87	0.73	0.80

Table 3: Results on topic lexicon extraction. Numbers in boldface denote significant improvement.

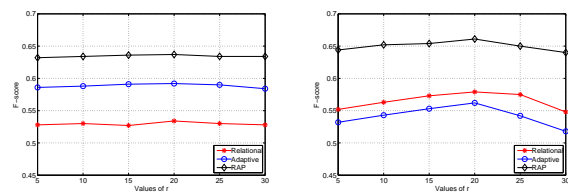
across domains, thus part of the labeled source domain data may be useful for the target learning task. However, for the topic lexicon extraction task, the topic words may be totally different, and as a result, we may not be able to find useful source domain labeled data to boost the performance for lexicon extraction in the target domain. In this case, mutual label propagation between sentiment and topic words may be more reasonable for knowledge transfer. RAP absorbs the advantages of the adaptive and relational bootstrapping methods, thus can get the best results in both lexicon extraction tasks.

We also observe that relational bootstrapping can get better recall, but lower precision, compared to adaptive bootstrapping. This is because relational bootstrapping only utilizes the patterns to propagate label information, which may cover more topic and sentiment seeds, but include some *noisy* words. For example, given two phases “like the camera” and “recommend the camera”, we can extract a pattern “VB-dobj-NN”. However, by using this pattern and the topic word “camera”, we may extract “take” as a sentiment word from another phase “take the cam-

era”, which is incorrect. The adaptive bootstrapping method can utilize various features to make predictions more precisely, which may have higher precision, but encounter the lower recall problem. For example, “flash” is not identified as a topic word in the target product domain (camera domain). Our RAP method can exploit both relationships between sentiment and topic words and part of labeled source domain data for cross-domain lexicon extraction. It can correctly identify the above two cases.

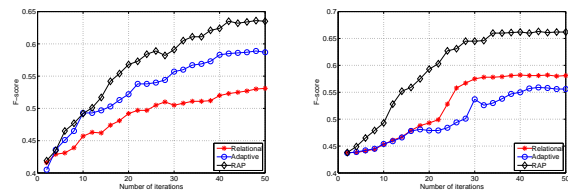
### 6.3.1 Parameter Sensitivity Study

In this section, we conduct experiments to study the effect of different parameter settings. There are several parameters in the framework: the number of generated seeds  $r$ , the number of new candidates  $k_2$  and the number of selections  $k$  in each iteration, and the number of iterations  $M$  ( $\mu$  is empirically set to 0.5). For the parameter  $k_2$ , we just set it to a large number ( $k_2 = 100$ ) such that have rich candidates to build the bipartite graph. In the experiments reported in the previous section, we set  $r = 20$ ,  $k_1 = 10$  and  $M = 50$ . Figures 4(a) and 4(b) show the results under varying values of  $r$  in the “product vs. movie” task. Observe that for sentiment word extraction, the results of the proposed methods are not sensitive to the values of  $r$ . While for the topic word extraction, the proposed methods perform well when  $r$  falls in the range from 15 to 20.



(a) Sentiment word extraction (b) Topic word extraction

Figure 4: Results on varying values of  $r$ .



(a) Sentiment word extraction (b) Topic word extraction

Figure 5: Results on varying values of  $M$ .

We also test the sensitivity of the parameter  $k_1$  and find that the proposed methods work well and robust when  $k_1$  falls in the range from 10 to 20.

Figures 5(a) and 5(b) show the results under varying numbers of iterations in the “product vs. movie” task. As we can see, our proposed methods converge well when  $M \geq 40$ .

## 7 Application: Sentiment Classification

To further verify the usefulness of the lexicons extracted by the RAP method, we apply the extracted sentiment lexicon for sentiment classification.

### 7.1 Experiment Setting

Our work is motivated by the work of (Pang and Lee, 2004), which only used subjective sentences for document-level sentiment classification, instead of using all sentences. In this experiment, we only use sentiment related words as features to represent opinion documents for classification, instead of using all words. Our goal is compare the sentiment lexicon constructed by the RAP method with other general lexicons on the impact of for sentiment classification. The general lexicons used for comparison are described in Table 4.

We use the dataset from (Blitzer et al., 2007) for sentiment classification. It contains a collection of product reviews from Amazon.com. The reviews are about four product domains: books, dvds, electronics and kitchen appliance. In each domain, there are 1000 positive and 1000 negative reviews. To construct domain specific sentiment lexicons, we apply RAP on each product domain with the *movie* domain described in Section 6.1 as the source domain. Finally, we use linear SVM as the classifier and the classification accuracy as the evaluate criterion.

Lexicon Name	Size	Description
Senti-WordNet	6957	Words with a subjective score $> 0.6$ (Esuli and Sebastiani, 2006)
HowNet	4619	Eng. translation of subj. Chinese words (Dong and Dong, 2006)
Subj. Clues	6878	Lexicons from (Wilson et al., 2005)

Table 4: Description of different lexicons.

### 7.2 Experimental Results

Experimental results on sentiment classification are shown in Table 5, where we denote “All” using all unigram and bigram features instead of using subjective words. As we can see that a classifier trained with features constructed by our RAP method performance best in all domains. Note that the number of features (sentiment words) constructed by our

method is much smaller than that of all unigram and bigram features, which can reduce the classifier training time dramatically. These promising results imply that our RAP can be applied for sentiment classification effectively and efficiently.

	All	Senti	HowNet	Subj. Clue	Ours
dvd	82.55	79.80	80.57	80.93	<b>84.05</b>
book	80.71	76.22	78.22	79.48	<b>81.65</b>
electronic	84.43	82.42	83.05	83.22	<b>86.71</b>
kitchen	87.70	81.78	84.17	84.23	<b>88.83</b>

Table 5: Sentiment classification results (accuracy in %). Numbers in boldface denotes significant improvement.

## 8 Conclusions

In this paper, we propose a two-stage framework for co-extraction of sentiment and topic lexicons across domains where we have no labeled data in the target domain but have plenty of labeled data in another domain. In the first stage, we propose a simple strategy to generate a few high-quality sentiment and topic seeds for the target domain. In the second stage, we propose a novel Relational Adaptive bootstrapping (RAP) method to expand the seeds, which can exploit the relationships between topic and opinion words, and make use of part of useful source domain labeled data for help. Extensive experimental results show our proposed method can extract precise sentiment and topic lexicons from the target domain. Furthermore, the extracted sentiment lexicon can be applied to sentiment classification effectively.

In the future work, besides the heterogeneous relationships between topic and sentiment words, we intend to investigate the homogeneous relationships among topic words and those among sentiment words (Qiu et al., 2009) to further boost the performance of RAP method. Furthermore, in our framework, we do not identify the polarity of the extracted sentiment lexicon. We also plan to embed this component into our unified framework. Finally, it is also interesting to exploit multi-domain knowledge (Li and Zong, 2008; Bollegala et al., 2011) for cross-domain lexicon extraction.

## 9 Acknowledgement

This work was supported by the Chinese Natural Science Foundation No.60973104, National Key Basic Research Program 2012CB316301, and Hong Kong RGC GRF Projects 621010 and 621211.

## References

- Rie K. Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 432–439, Prague, Czech Republic. ACL.
- Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 92–100.
- Danushka Bollegala, David Weir, and John Carroll. 2011. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 132–141, Portland, Oregon. ACL.
- Wenyuan Dai, Qiang Yang, Guirong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning*, pages 193–200, Corvallis, Oregon, USA, June. ACM.
- Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. ACL.
- Zhendong Dong and Qiang Dong, editors. 2006. *HOWNET and the computation of meaning*. World Scientific Publishers, Norwell, MA, USA.
- Weifu Du, Songbo Tan, Xueqi Cheng, and Xiaochun Yun. 2010. Adapting information bottleneck method for automatic construction of domain-oriented sentiment lexicon. In *Proceedings of the 3rd ACM international conference on Web search and data mining*, pages 111–120, New York, NY, USA. ACM.
- Andrea Esuli and Fabrizio Sebastiani. 2006. SENTIWORDNET: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation*, pages 417–422.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning*, pages 513–520, Bellevue, Washington, USA.
- Yulan He, Chenghua Lin, and Harith Alani. 2011. Automatically extracting polarity-bearing topics for cross-domain sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 123–131, Portland, Oregon. ACL.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, Seattle, WA, USA. ACM.
- Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1035–1045, Cambridge, Massachusetts, USA. ACL.
- Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic. ACL.
- Wei Jin and Hung Hay Ho. 2009. A novel lexicalized HMM-based learning framework for web opinion mining. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 465–472, Montreal, Quebec, Canada. ACM.
- Rosie Jones, Andrew McCallum, Kamal Nigam, and Ellen Riloff. 1999. Bootstrapping for text learning tasks. In *In IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications*, pages 52–63.
- Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *J. ACM*, 46:604–632, Sept.
- Shoushan Li and Chengqing Zong. 2008. Multi-domain sentiment classification. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 257–260, Columbus, Ohio, USA. ACL.
- Tao Li, Vikas Sindhwani, Chris Ding, and Yi Zhang. 2009. Knowledge transformation for cross-domain sentiment classification. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 716–717, Boston, MA, USA. ACM.
- Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010a. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 653–661, Beijing, China.
- Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010b. Sentiment analysis with global topics and local dependency. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, Atlanta, Georgia, USA. AAAI Press.

- Bing Liu. 2010. Sentiment analysis and subjectivity. *Handbook of Natural Language Processing, Second Edition*.
- Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of the 16th international conference on World Wide Web*, pages 171–180, Banff, Alberta, Canada. ACM.
- Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, Oct.
- Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Chen Zheng. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th International Conference on World Wide Web*, pages 751–760, Raleigh, NC, USA, Apr. ACM.
- Bo Pang and Lillian Lee. 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Barcelona, Spain. ACL.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135.
- Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 339–346, Vancouver, British Columbia, Canada. ACL.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international joint conference on Artificial intelligence*, pages 1199–1204, Pasadena, California, USA. Morgan Kaufmann Publishers Inc.
- Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the 6th national conference on Artificial intelligence*, pages 474–479, Orlando, Florida, United States. AAAI.
- Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the 7th conference on natural language learning*, pages 25–32, Edmonton, Canada. ACL.
- Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1044–1049, Portland, Oregon, USA. AAAI Press/MIT Press.
- Songbo Tan, Gaowei Wu, Huifeng Tang, and Xueqi Cheng. 2007. A novel scheme for domain-transfer problem in the context of sentiment analysis. In *Proceedings of the 16th ACM conference on Conference on information and knowledge management*, pages 979–982, Lisbon, Portugal. ACM.
- Ivan Titov and Ryan McDonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics: Human Language Technologies*, pages 308–316, Columbus, Ohio, USA. ACL.
- Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Comput. Linguist.*, 30:277–308, Sept.
- Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354, Vancouver, British Columbia, Canada. ACL.
- Dan Wu, Wee Sun Lee, Nan Ye, and Hai Leong Chieu. 2009. Domain adaptive bootstrapping for named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1523–1532, Singapore. ACL.
- Min Zhang and Xingyao Ye. 2008. A generation model to unify topic relevance and lexicon-based sentiment for opinion retrieval. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 411–418, Singapore. ACM.
- Wayne Xin Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. 2010. Jointly modeling aspects and opinions with a MaxEnt-LDA hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 56–65, Cambridge, Massachusetts, USA. ACL.

# Learning Syntactic Verb Frames Using Graphical Models

**Thomas Lippincott**  
University of Cambridge  
Computer Laboratory  
United Kingdom  
t1318@cam.ac.uk

**Diarmuid Ó Séaghdha**  
University of Cambridge  
Computer Laboratory  
United Kingdom  
do242@cam.ac.uk

**Anna Korhonen**  
University of Cambridge  
Computer Laboratory  
United Kingdom  
alk23@cam.ac.uk

## Abstract

We present a novel approach for building verb subcategorization lexicons using a simple graphical model. In contrast to previous methods, we show how the model can be trained without parsed input or a predefined subcategorization frame inventory. Our method outperforms the state-of-the-art on a verb clustering task, and is easily trained on arbitrary domains. This quantitative evaluation is complemented by a qualitative discussion of verbs and their frames. We discuss the advantages of graphical models for this task, in particular the ease of integrating semantic information about verbs and arguments in a principled fashion. We conclude with future work to augment the approach.

## 1 Introduction

Subcategorization frames (SCFs) give a compact description of a verb’s syntactic preferences. These two sentences have the same sequence of lexical syntactic categories (VP-NP-SCOMP), but the first is a simple transitive (“X understood Y”), while the second is a ditransitive with a sentential complement (“X persuaded Y that Z”):

1. Kim (VP understood (NP the evidence (SCOMP that Sandy was present)))
2. Kim (VP persuaded (NP the judge) (SCOMP that Sandy was present))

An SCF lexicon would indicate that “persuade” is likely to take a direct object and sentential complement (NP-SCOMP), while “understand” is more likely to take just a direct object (NP). A comprehensive lexicon would also include semantic information about selectional preferences (or restrictions) on argument heads of verbs, diathesis alternations (i.e. semantically-motivated alternations between pairs of SCFs) and a mapping from surface frames to the underlying predicate-argument structure. Information about verb subcategorization is useful for tasks like information extraction (Cohen and Hunter, 2006; Rupp et al., 2010), verb clustering (Korhonen et al., 2006b; Merlo and Stevenson, 2001) and parsing (Carroll et al., 1998). In general, tasks that depend on predicate-argument structure can benefit from a high-quality SCF lexicon (Surdeanu et al., 2003).

Large, manually-constructed SCF lexicons mostly target general language (Boguraev and Briscoe, 1987; Grishman et al., 1994). However, in many domains verbs exhibit different syntactic behavior (Roland and Jurafsky, 1998; Lippincott et al., 2010). For example, the verb “develop” has specific usages in newswire, biomedicine and engineering that dramatically change its probability distribution over SCFs. In a few domains like biomedicine, the need for focused SCF lexicons has led to manually-built resources (Bodenreider, 2004). Such resources, however, are costly, prone to human error, and in domains where new lexical and syntactic constructs are frequently coined, quickly become obsolete (Cohen and Hunter, 2006). Data-driven methods for SCF acquisition can alleviate



these problems by building lexicons tailored to new domains with less manual effort, and higher coverage and scalability.

Unfortunately, high quality SCF lexicons are difficult to build automatically. The argument-adjunct distinction is challenging even for humans, many SCFs have no reliable cues in data, and some SCFs (e.g. those involving control such as type raising) rely on semantic distinctions. As SCFs follow a Zipfian distribution (Korhonen et al., 2000), many genuine frames are also low in frequency. State-of-the-art methods for building data-driven SCF lexicons typically rely on parsed input (see section 2). However, the treebanks necessary for training a high-accuracy parsing model are expensive to build for new domains. Moreover, while parsing may aid the detection of some frames, many experiments have also reported SCF errors due to noise from parsing (Korhonen et al., 2006a; Preiss et al., 2007).

Finally, many SCF acquisition methods operate with predefined SCF inventories. This subscribes to a single (often language or domain-specific) interpretation of subcategorization *a priori*, and ignores the ongoing debate on how this interpretation should be tailored to new domains and applications, such as the more prominent role of adjuncts in information extraction (Cohen and Hunter, 2006).

In this paper, we describe and evaluate a novel probabilistic data-driven method for SCF acquisition aimed at addressing some of the problems with current approaches. In our model, a Bayesian network describes how verbs choose their arguments in terms of a small number of frames, which are represented as distributions over syntactic relationships. First, we show that by allowing the inference process to automatically define a probabilistic SCF inventory, we outperform systems with hand-crafted rules and inventories, using identical syntactic features. Second, by replacing the syntactic features with an approximation based on POS tags, we achieve state-of-the-art performance without relying on error-prone unlexicalized or domain-specific lexicalized parsers. Third, we highlight a key advantage of our method compared to previous approaches: the ease of integrating and performing joint inference of additional syntactic and semantic information. We describe how we plan to exploit this in our future research.

## 2 Previous work

Many state-of-the-art SCF acquisition systems take *grammatical relations* (GRs) as input. GRs express binary dependencies between lexical items, and many parsers produce them as output, with some variation in inventory (Briscoe et al., 2006; De Marneffe et al., 2006). For example, a subject-relation like “ncsubj(HEAD, DEPENDENT)” expresses the fact that the lexical item referred to by HEAD (such as a present-tense verb) has the lexical item referred to by DEPENDENT as its subject (such as a singular noun). GR inventories include direct and indirect objects, complements, conjunctions, among other relations. The dependency relationships included in GRs correspond closely to the head-complement structure of SCFs, which is why they are the natural choice for SCF acquisition.

There are several SCF lexicons for general language, such as ANLT (Boguraev and Briscoe, 1987) and COMLEX (Grishman et al., 1994), that depend on manual work. VALEX (Preiss et al., 2007) provides SCF distributions for 6,397 verbs acquired from a parsed general language corpus via a system that relies on hand-crafted rules. There are also resources which provide information about both syntactic and semantic properties of verbs: VerbNet (Kipper et al., 2008) draws on several hand-built and semi-automatic sources to link the syntax and semantics of 5,726 verbs. FrameNet (Baker et al., 1998) provides semantic frames and annotated example sentences for 4,186 verbs. PropBank (Palmer et al., 2005) is a corpus where each verb is annotated for its arguments and their semantic roles, covering a total of 4,592 verbs.

There are many language-specific SCF acquisition systems, e.g. for French (Messiant, 2008), Italian (Lenci et al., 2008), Turkish (Han et al., 2008) and Chinese (Han et al., 2008). These typically rely on language-specific knowledge, either directly through heuristics, or indirectly through parsing models trained on treebanks. Furthermore, some require labeled training instances for supervised (Uzun et al., 2008) or semi-supervised (Han et al., 2008) learning algorithms.

Two state-of-the-art data-driven systems for English verbs are those that produced VALEX, Preiss et al. (2007), and the BioLexicon (Venturi et al., 2009).

The Preiss system extracts a verb instance’s GRs using the Rasp general-language unlexicalized parser (Briscoe et al., 2006) as input, and based on hand-crafted rules, maps verb instances to a predefined inventory of 168 SCFs. Filtering is then performed to remove noisy frames, with methods ranging from a simple single threshold to SCF-specific hypothesis tests based on external verb classes and SCF inventories. The BioLexicon system extracts each verb instance’s GRs using the lexicalized Enju parser tuned to the biomedical domain (Miyao, 2005). Each unique GR-set considered a potential SCF, and an experimentally-determined threshold is used to filter low-frequency SCFs.

Note that both methods require extensive manual work: the Preiss system involves the *a priori* definition of the SCF inventory, careful construction of matching rules, and an unlexicalized parsing model. The BioLexicon system induces its SCF inventory automatically, but requires a lexicalized parsing model, rendering it more sensitive to domain variation. Both rely on a filtering stage that depends on external resources and/or gold standards to select top-performing thresholds. Our method, by contrast, does not use a predefined SCF inventory, and can perform well without parsed input.

Graphical models have been increasingly popular for a variety of tasks such as distributional semantics (Blei et al., 2003) and unsupervised POS tagging (Finkel et al., 2007), and sampling methods allow efficient estimation of full joint distributions (Neal, 1993). The potential for joint inference of complementary information, such as syntactic verb and semantic argument classes, has a clear and interpretable way forward, in contrast to the pipelined methods described above. This was demonstrated in Andrew et al. (2004), where a Bayesian model was used to jointly induce syntactic and semantic classes for verbs, although that study relied on manually annotated data and a predefined SCF inventory and MLE. More recently, Abend and Rappoport (2010) trained ensemble classifiers to perform argument-adjunct disambiguation of PP complements, a task closely related to SCF acquisition. Their study employed unsupervised POS tagging and parsing, and measures of selectional preference and argument structure as complementary features for the classifier.

Finally, our task-based evaluation, verb clustering with Levin (1993)’s alternation classes as the gold standard, was previously conducted by Joanis and Stevenson (2003), Korhonen et al. (2008) and Sun and Korhonen (2009).

### 3 Methodology

In this section we describe the basic components of our study: feature sets, graphical model, inference, and evaluation.

#### 3.1 Input and feature sets

We tested several feature sets either based on, or approximating, the concept of *grammatical relation* described in section 2. Our method is agnostic regarding the exact definition of GR, and for example could use the Stanford inventory (De Marneffe et al., 2006) or even an entirely different lexico-syntactic formalism like CCG supertags (Curran et al., 2007). In this paper, we distinguish “true GRs” (tGRs), produced by a parser, and “pseudo GRs” (pGRs), a POS-based approximation, and employ subscripts to further specify the variations described below. Our input has been parsed into Rasp-style tGRs (Briscoe et al., 2006), which facilitates comparison with previous work based on the same data set.

We’ll use a simple example sentence to illustrate how our feature sets are extracted from CONLL-formatted data (Nivre et al., 2007). The CONLL format is a common language for comparing output from dependency parsers: each lexical item has an index, lemma, POS tag, tGR in which it is the dependent, and index to the corresponding head. Table 1 shows the relevant fields for the sentence “We run training programmes in Romania and other countries”.

We define the feature set for a verb occurrence as the counts of each GR the verb participates in. Table 2 shows the three variations we tested: the simple tGR type, with parameterization for the POS tags of head and dependent, and with closed-class POS tags (determiners, pronouns and prepositions) lexicalized. In addition, we tested the effect of limiting the features to subject, object and complement tGRs, indicated by adding the subscript “lim”, for a total of six tGR-based feature sets.

While ideally tGRs would give full informa-

Index	Lemma	POS	Head	tGR
1	we	PPIS2	2	ncsubj
2	run	VV0	0	-
3	training	NN1	4	ncmod
4	programme	NN2	2	doobj
5	in	II	4	ncmod
6	romania	NP1	7	conj
7	and	CC	5	doobj
8	other	JB	9	ncmod
9	country	NN2	7	conj

Table 1: Simplified CONLL format for example sentence “We run training programmes in Romania and other countries”. Head=0 indicates the token is the root.

Name	Features	
$tGR$	ncsubj	doobj
$tGR_{param}$	ncsubj(VV0,PPIS2)	doobj(VV0,NN2)
$tGR_{param,lex}$	ncsubj(VV0,PPIS2-we)	doobj(VV0,NN2)

Table 2: True-GR features for example sentence: note there are also  $tGR_{*,lim}$  versions of each that only consider subjects, objects and complements and are not shown.

tion about the verb’s syntactic relationship to other words, in practice parsers make (possibly premature) decisions, such as deciding that “in” modifies “programme”, and not “run” in our example sentence. An unlexicalized parser cannot distinguish these based just on POS tags, while a lexicalized parser requires a large treebank. We therefore define *pseudo-GRs* (pGRs), which consider each (distance, POS) pair within a given window of the verb to be a potential tGR. Table 3 shows the pGR features for the test sentence using a window of three. As with tGRs, the closed-class tags can be lexicalized, but there are no corresponding feature sets for *param* (since they are already built from POS tags) or *lim* (since there is no similar rule-based approach).

Name	Features			
$pGR$	-1(PPIS2)	1(NN1)	2(NN2)	3(II)
$pGR_{lex}$	-1(PPIS2-we)	1(NN1)	2(NN2)	3(II-in)

Table 3: Pseudo-GR features for example sentence with window=3

Whichever feature set is used, an instance is sim-

ply the count of each GR’s occurrences. We extract instances for the 385 verbs in the union of our two gold standards from the VALEX lexicon’s data set, which was used in previous studies (Sun and Korhonen, 2009; Preiss et al., 2007) and facilitates comparison with that resource. This data set is drawn from five general-language corpora parsed by Rasp, and provides, on average, 7,000 instances per verb.

### 3.2 SCF extraction

Our graphical modeling approach uses the Bayesian network shown in Figure 1. Its generative story is as follows: when a verb is instantiated, an SCF is chosen according to a verb-specific multinomial. Then, the number and type of syntactic arguments (GRs) are chosen from two SCF-specific multinomials. These three multinomials are modeled with uniform Dirichlet priors and corresponding hyperparameters  $\alpha$ ,  $\beta$  and  $\gamma$ . The model is trained via collapsed Gibbs sampling, where the probability of assigning a particular SCF  $s$  to an instance of verb  $v$  with GRs ( $gr_1 \dots gr_n$ ) is the product

$$\begin{aligned}
 P(s|Verb = v, GRs = gr_1 \dots gr_n) = & \\
 & P(SCF = s|Verb = v) \times \\
 & P(N = n|SCF = s) \times \\
 & \prod_{i=1:n} P(GR = gr_i|SCF = s)
 \end{aligned}$$

The three terms, given the hyper-parameters and conjugate-prior relationship between Dirichlet and Multinomial distributions, can be expressed in terms of current assignments of  $s$  to verb  $v$  ( $c_{sv}$ ),  $s$  to GR-count  $n$  ( $c_{sn}$ ) and  $s$  to GR ( $c_{sg}$ ), the corresponding totals ( $c_v$ ,  $c_s$ ), the dimensionality of the distributions ( $|SCF|$ ,  $|N|$  and  $|G|$ ) and the hyperparameters  $\alpha$ ,  $\beta$  and  $\gamma$ :

$$P(SCF = s|Verb = v) = (c_{sv} + \alpha) / (c_v + |SCF|\alpha)$$

$$P(N = n|SCF = s) = (c_{sn} + \beta) / (c_s + |N|\beta)$$

$$P(GR = gr_i|SCF = s) = (c_{sgr_i} + \gamma) / (c_s + |G|\gamma)$$

Note that  $N$ , the possible GR-count for an instance, is usually constant for pGRs ( $2 \times window$ ), unless the verb is close to the start or end of the sentence.

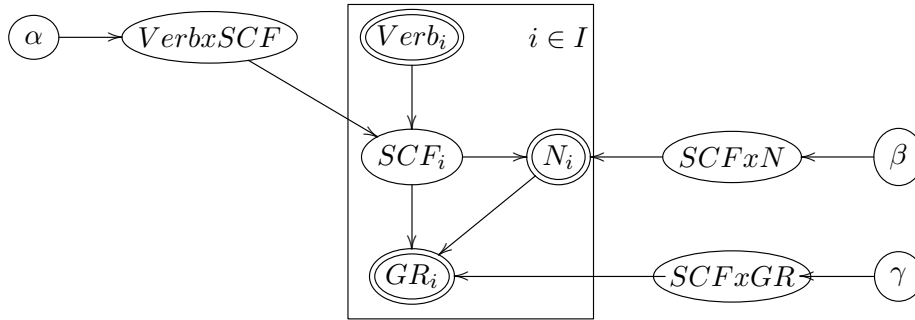


Figure 1: Our simple graphical model reflecting subcategorization. Double-circles indicate an observed value, arrows indicate conditional dependency. What constitutes a “GR” depends on the feature set being used.

We chose our hyper-parameters  $\alpha = \beta = \gamma = .02$  to reflect the characteristic sparseness of the phenomena (i.e. verbs tend to take a small number of SCFs, which in turn are limited to a small number of realizations). For the pGRs we used a window of 5 tokens: a verb’s arguments will fall within a small window in the majority of cases, so there is diminished return in expanding the window at the cost of increased noise. Finally, we set our SCF count to 40, about twice the size of the strictly syntactic general-language gold standard we describe in section 3.3. This overestimation allows some flexibility for the model to define its inventory based on the data; any supernumerary frames will act as “junk frames” that are rarely assigned and hence will have little influence. We run Gibbs sampling for 1000 iterations, and average the final 100 samples to estimate the posteriors  $P(SCF|Verb)$  and  $P(GR|SCF)$ . Variance between adjacent states’ estimates of  $P(SCF|Verb)$  indicates that the sampling typically converges after about 100-200 iterations.<sup>1</sup>

### 3.3 Evaluation

#### Quantitative: cluster gold standard

Evaluating the output of unsupervised methods is not straightforward: discrete, expert-defined categories (like many SCF inventories) are unlikely to line up perfectly with data-driven, probabilistic output. Even if they do, finding a mapping between them is a problem of its own (Meila, 2003).

<sup>1</sup>Full source code for this work is available at <http://cl.cam.ac.uk/~t1318/files/subcat.tgz>

Our goal is to define a fair quantitative comparison between arbitrary SCF lexicons. An SCF lexicon makes two claims: first, that it defines a reasonable SCF inventory. Second, that for each verb, it has an accurate distribution over that inventory. We therefore compare the lexicons based on their performance on a task that a good SCF lexicon should be useful for: clustering verbs into lexical-semantic classes. Our gold standard is from (Sun and Korhonen, 2009), where 200 verbs were assigned to 17 classes based on their alternation patterns (Levin, 1993). Previous work (Schulte im Walde, 2009; Sun and Korhonen, 2009) has demonstrated that the quality of an SCF lexicon’s inventory and probability estimates corresponds to its predictive power for membership in such alternation classes.

To compare the performance of our feature sets, we chose the simple and familiar K-Means clustering algorithm (Hartigan and Wong, 1979). The instances are the verbs’ SCF distributions, and we select the number of clusters by the Silhouette validation technique (Rousseeuw, 1987). The clusters are then compared to the gold standard clusters with the purity-based F-Score from Sun and Korhonen (2009) and the more familiar Adjusted Rand Index (Hubert and Arabie, 1985). Our main point of comparison is the VALEX lexicon of SCF distributions, whose scores we report alongside ours.

#### Qualitative: manual gold standard

We also want to see how our results line up with a traditional linguistic view of subcategorization, but this requires digging into the unsupervised out-

put and associating anonymous probabilistic objects with established categories. We therefore present sample output in three ways: first, we show the clustering output from our top-performing method. Second, we plot the probability mass over GRs for two anonymous SCFs that correspond to recognizable traditional SCFs, and one that demonstrates unexpected behavior. Third, we compared the output for several verbs to a coarsened version of the manually-annotated gold standard used to evaluate VALEX (Preiss et al., 2007). We collapsed the original inventory of 168 SCFs to 18 purely syntactic SCFs based on their characteristic GRs and removed frames that depend on semantic distinctions, leaving the detection of finer-grained and semantically-based frames for future work.

## 4 Results

### 4.1 Verb clustering

We evaluated SCF lexicons based on the eight feature sets described in section 3.1, as well as the VALEX SCF lexicon described in section 2. Table 4 shows the performance of the lexicons in ascending order.

Method	Pur. F-score	Adj. Rand
<i>tGR</i>	.24	.02
<i>tGR<sub>lim</sub></i>	.27	.02
<i>pGR<sub>lex</sub></i>	.32	.09
<i>tGR<sub>lim,param</sub></i>	.35	.08
<i>pGR</i>	.35	.10
VALEX	.36	.10
<i>tGR<sub>param,lex</sub></i>	.37	.10
<i>tGR<sub>param</sub></i>	.39	.12
<i>tGR<sub>lim,param,lex</sub></i>	.44	.12

Table 4: Task-based evaluation of lexicons acquired with each of the eight feature types, and the state-of-the-art rule-based VALEX lexicon.

These results lead to several conclusions: first, training our model on tGRs outperforms pGRs and VALEX. Since the parser that produced them is known to perform well on general language (Briscoe et al., 2006), the tGRs are of high quality: it makes sense that reverting to the pGRs is unnecessary in this case. The interesting point is the major performance gain over VALEX, which uses the same tGR

features along with expert-developed rules and inventory.

Second, we achieve performance comparable to VALEX using pGRs with a narrow window width. Since POS tagging is more reliable and robust across domains than parsing, retraining on new domains will not suffer the effects of a mismatched parsing model (Lippincott et al., 2010). It is therefore possible to use this method to build large-scale lexicons for any new domain with sufficient data.

Third, lexicalizing the closed-class POS tags introduces semantic information outside the scope of the alternation-based definition of subcategorization. For example, subdividing the indefinite pronoun tag “PN1” into “PN1-anyone” and “PN1-anything” gives information about the animacy of the verb’s arguments. Our results show this degrades performance for both pGR and tGR features, unless the latter are limited to tGRs traditionally thought to be relevant for the task.

### 4.2 Qualitative analysis

Table 5 shows clusters produced by our top-scoring method,  $GR_{param,lex,lim}$ . Some clusters are immediately intelligible at the semantic level and correspond closely to the lexical-semantic classes found in Levin (1993). For example, clusters 1, 6, and 14 include member verbs of Levin’s SAY, PEER and AMUSE classes, respectively. Some clusters are based on broader semantic distinctions (e.g. cluster 2 which groups together verbs related to locations) while others relate semantic classes purely based on their syntactic similarity (e.g. the verbs in cluster 17 share strong preference for ‘to’ preposition). The syntactic-semantic nature of the clusters reflects the multimodal nature of verbs and illustrates why a comprehensive subcategorization lexicon should not be limited to syntactic frames. This phenomenon is also encouraging for future work to tease apart and simultaneously exploit several verbal aspects via additional latent structure in the model.

An SCF’s distribution over features can reveal its place in the traditional definition of subcategorization. Figure 2 shows the high-probability (>.02) tGRs for one SCF: the large mass centered on direct object tGRs indicates this approximates the notion of “transitive”. Looking at the verbs most likely to take this SCF (“stimulate”, “conserve”) confirms

1	exclaim, murmur, mutter, reply, retort, say, sigh, whisper
2	bang, knock, snoop, swim, teeter
3	flicker, multiply, overlap, shine
4	batter, charter, compromise, overwhelm, regard, sway, treat
5	abolish, broaden, conserve, deepen, eradicate, remove, sharpen, shorten, stimulate, strengthen, unify
6	gaze, glance, look, peer, sneer, squint, stare
7	coincide, commiserate, concur, flirt, interact
8	grin, smile, wiggle
9	confuse, diagnose, march
10	mate, melt, swirl
11	frown, jog, stutter
12	chuckle, mumble, shout
13	announce, envisage, mention, report, state
14	frighten, intimidate, scare, shock, upset
15	bash, falter, snarl, wail, weaken
16	cooperate, eject, respond, transmit
17	affiliate, compare, contrast, correlate, forward, mail, ship

Table 5: Clusters (of size  $>2$  and  $<20$ ) produced using  $tGR_{param,lex,lim}$

this. Figure 3 shows a complement-taking SCF, which is far rarer than simple transitive but also clearly induced by our model.

The induced SCF inventory also has some redundancy, such as additional transitive frames beside figure 2, and frames with poor probability estimates. Most of these issues can be traced to our simplifying assumption that each tGR is drawn independently w.r.t. an instance’s other tGRs. For example, if an SCF gives *any* weight to indirect objects, it gives non-zero probability to an instance with *only* indirect objects, an impossible case. This can lead to skewed probability estimates: since some tGRs can occur multiple times in a given instance (e.g. indirect objects and prepositional phrases) the model may find it reasonable to create an SCF with all probability focused on that tGR, ignoring all others, such as in figure 4. We conclude that our independence assumption was too strong, and the model would benefit from defining more structure within

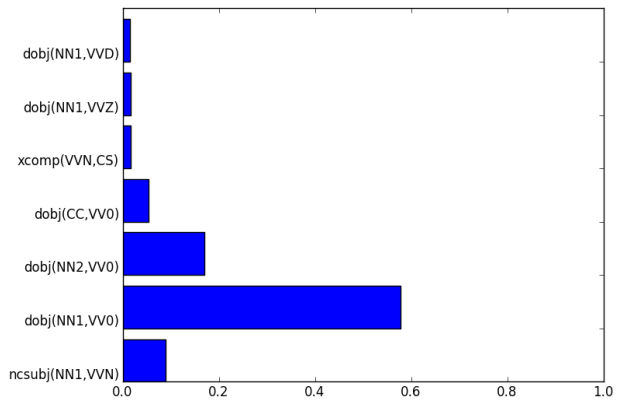


Figure 2: The SCF corresponding to transitive has most probability centered on dobj (e.g. stimulate, conserve, deepen, eradicate, broaden)

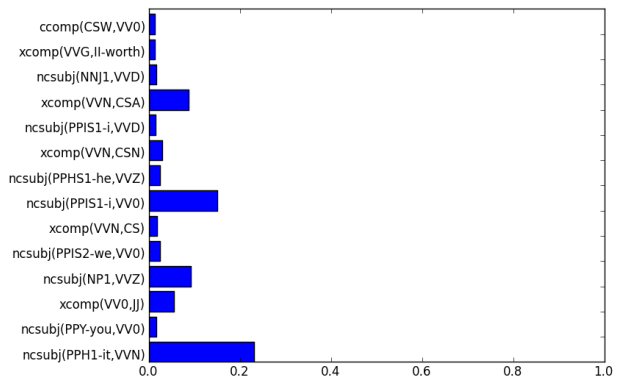


Figure 3: The SCF corresponding to verbs taking complements has more probability on xcomp and ccomp (e.g. believe, state, agree, understand, mention)

instances.

The full tables necessary to compare verb SCF distributions from our output with the manual gold standard are prohibited by space, but a few examples reinforce the analysis above. The verbs “load” and “fill” show particularly high usage of ditransitive SCFs in the gold standard. In our inventory, this is reflected in high usage of an SCF with probability centered on indirect objects, but due to the independence assumptions the frame has a corresponding low probability on subjects and direct objects, despite the fact that these necessarily occur *along with* any indirect object. The verbs “acquire” and “buy” demonstrate both a strength of our approach and a weakness of using parsed input: both verbs

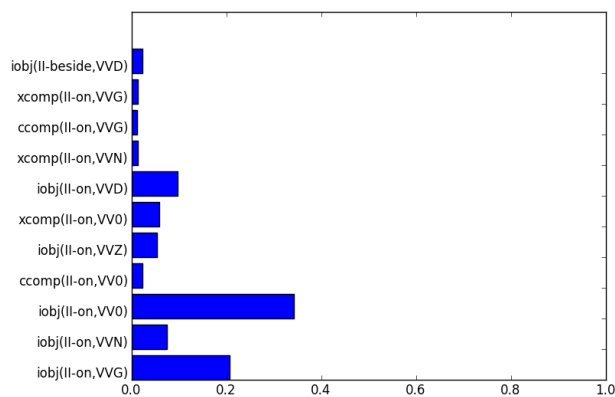


Figure 4: This SCF is dominated by indirect objects and complements, catering to verbs that may take several such tGRs, at the expense of subjects

show high probability of simple transitive in our output and the gold standard. However, the Rasp parser often conflates indirect objects and prepositional phrases due to its unlexicalized model. While our system correctly gives high probability to ditransitive for both verbs, it inherits this confusion and over-estimates “acquire”’s probability mass for the frame. This is an example of how bad decisions made by the parser cannot be fixed by the graphical model, and an area where pGR features have an advantage.

## 5 Conclusions and future work

Our study reached two important conclusions: first, given the same data as input, an unsupervised probabilistic model can outperform a hand-crafted rule-based SCF extractor with a predefined inventory. We achieve better results with far less effort than previous approaches by allowing the data to govern the definition of frames while estimating the verb-specific distributions in a fully Bayesian manner. Second, simply treating POS tags within a small window of the verb as pseudo-GRs produces state-of-the-art results without the need for a parsing model. This is particularly encouraging when building resources for new domains, where complex models fail to generalize. In fact, by integrating results from unsupervised POS tagging (Teichert and Daumé III, 2009) we could render this approach fully domain- and language-independent.

We did not dwell on issues related to choosing

our hyper-parameters or latent class count. Both of these can be accomplished with additional sampling methods: hyper-parameters of Dirichlet priors can be estimated via slice sampling (Heinrich, 2009), and their dimensionality via Dirichlet Process priors (Heinrich, 2011). This could help address the redundancy we find in the induced SCF inventory, with the potential SCFs growing to accommodate the data.

Our initial attempt at applying graphical models to subcategorization also suggested several ways to extend and improve the method. First, the independence assumptions between GRs in a given instance turned out to be too strong. To address this, we could give instances internal structure to capture conditional probability between generated GRs. Second, our results showed the conflation of several verbal aspects, most notably the syntactic and semantic. In a sense this is encouraging, as it motivates our most exciting future work: augmenting this simple model to explicitly capture complementary information such as distributional semantics (Blei et al., 2003), diathesis alternations (McCarthy, 2000) and selectional preferences (Ó Séaghdha, 2010). This study targeted high-frequency verbs, but the use of syntactic and semantic classes would also help with data sparsity down the road. These extensions would also call for a more comprehensive evaluation, averaging over several tasks, such as clustering by semantics, syntax, alternations and selectional preferences.

In concrete terms, we plan to introduce latent variables corresponding to syntactic, semantic and alternation classes, that will determine a verb’s syntactic arguments, their semantic realization (i.e. selectional preferences), and possible predicate-argument structures. By combining the syntactic classes with unsupervised POS tagging (Teichert and Daumé III, 2009) and the selectional preferences with distributional semantics (Ó Séaghdha, 2010), we hope to produce more accurate results on these complementary tasks while avoiding the use of any supervised learning. Finally, a fundamental advantage of a data-driven, parse-free method is that it can be easily trained for new domains. We next plan to test our method on a new domain, such as biomedical text, where verbs are known to take on distinct syntactic behavior (Lippincott et al., 2010).

## 6 Acknowledgements

The work in this paper was funded by the Royal Society, (UK), EPSRC (UK) grant EP/G051070/1 and EU grant 7FP-ITC-248064. We are grateful to Lin Sun and Laura Rimell for the use of their clustering and subcategorization gold standards, and the ACL reviewers for their helpful comments and suggestions.

## References

- Omri Abend and Ari Rappoport. 2010. Fully unsupervised core-adjunct argument classification. In *ACL '10*.
- Galen Andrew, Trond Grenager, and Christopher Manning. 2004. Verb sense and subcategorization: using joint inference to improve performance on complementary tasks. *EMNLP '04*.
- Collin Baker, Charles Fillmore, and John Lowe. 1998. The Berkeley FrameNet project. In *COLING ACL '98*.
- David Blei, Andrew Ng, Michael Jordan, and John Lafferty. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.
- Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32.
- Bran Boguraev and Ted Briscoe. 1987. Large lexicons for natural language processing. *Computational Linguistics*, 13.
- Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the COLING/ACL on Interactive presentation sessions*.
- John Carroll, Guido Minnen, and Ted Briscoe. 1998. Can subcategorisation probabilities help a statistical parser? In *The 6th ACL/SIGDAT Workshop on Very Large Corpora*.
- K Bretonnel Cohen and Lawrence Hunter. 2006. A critical review of PASBio's argument structures for biomedical verbs. *BMC Bioinformatics*, 7.
- James Curran, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-Scale NLP with C&C and Boxer. In *ACL '07*.
- Marie-Catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC '06*.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2007. The infinite tree. In *ACL '07*.
- Ralph Grishman, Catherine Macleod, and Adam Meyers. 1994. Complex syntax: building a computational lexicon. In *COLING '94*.
- Xiwu Han, Chengguo Lv, and Tiejun Zhao. 2008. Weakly supervised SVM for Chinese-English cross-lingual subcategorization lexicon acquisition. In *The 11th Joint Conference on Information Science*.
- J.A. Hartigan and M.A. Wong. 1979. Algorithm AS 136: A K-Means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*.
- Gregor Heinrich. 2009. Parameter estimation for text analysis. Technical report, Fraunhofer IGD.



- Gregor Heinrich. 2011. Infinite LDA implementing the HDP with minimum code complexity. Technical report, arbylon.net.
- Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of Classification*, 2.
- Eric Joanis and Suzanne Stevenson. 2003. A general feature space for automatic verb classification. In *EACL '03*.
- Karin Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2008. A large-scale classification of English verbs. In *LREC '08*.
- Anna Korhonen, Genevieve Gorrell, and Diana McCarthy. 2000. Statistical filtering and subcategorization frame acquisition. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.
- Anna Korhonen, Yuval Krymolowski, and Ted Briscoe. 2006a. A large subcategorization lexicon for natural language processing applications. In *LREC '06*.
- Anna Korhonen, Yuval Krymolowski, and Nigel Collier. 2006b. Automatic classification of verbs in biomedical texts. In *ACL '06*.
- Anna Korhonen, Yuval Krymolowski, and Nigel Collier. 2008. The choice of features for classification of verbs in biomedical texts. In *COLING '08*.
- Ro Lenci, Barbara McGillivray, Simonetta Montemagni, and Vito Pirrelli. 2008. Unsupervised acquisition of verb subcategorization frames from shallow-parsed corpora. In *LREC '08*.
- Beth Levin. 1993. *English Verb Classes and Alternation: A Preliminary Investigation*. University of Chicago Press, Chicago, IL.
- Thomas Lippincott, Anna Korhonen, and Diarmuid Ó Séaghdha. 2010. Exploring subdomain variation in biomedical language. *BMC Bioinformatics*.
- Diana McCarthy. 2000. Using semantic preferences to identify verbal participation in role switching alternations. In *NAACL '00*.
- Marina Meila. 2003. Comparing clusterings by the Variation of Information. In *COLT*.
- Paola Merlo and Suzanne Stevenson. 2001. Automatic verb classification based on statistical distributions of argument structure. *Computational Linguistics*.
- Cédric Messiant. 2008. A subcategorization acquisition system for French verbs. In *ACL HLT '08 Student Research Workshop*.
- Yusuke Miyao. 2005. Probabilistic disambiguation models for wide-coverage HPSG parsing. In *ACL '05*.
- Radford M. Neal. 1993. Probabilistic inference using markov chain Monte Carlo methods. Technical report, University of Toronto.
- Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *The CoNLL Shared Task Session of EMNLP-CoNLL 2007*.
- Diarmuid Ó Séaghdha. 2010. Latent variable models of selectional preference. In *ACL '10*.
- Martha Palmer, Paul Kingsbury, and Daniel Gildea. 2005. The Proposition Bank: an annotated corpus of semantic roles. *Computational Linguistics*.
- Judita Preiss, Ted Briscoe, and Anna Korhonen. 2007. A system for large-scale acquisition of verbal, nominal and adjectival subcategorization frames from corpora. In *ACL '07*.
- Douglas Roland and Daniel Jurafsky. 1998. How verb subcategorization frequencies are affected by corpus choice. In *ACL '98*.
- Peter Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*.
- C.J. Rupp, Paul Thompson, William Black, and John McNaught. 2010. A specialised verb lexicon as the basis of fact extraction in the biomedical domain. In *Interdisciplinary Workshop on Verbs: The Identification and Representation of Verb Features*.
- Sabine Schulte im Walde. 2009. The induction of verb frames and verb classes from corpora. In *Corpus Linguistics. An International Handbook*. Mouton de Gruyter.
- Lin Sun and Anna Korhonen. 2009. Improving verb clustering with automatically acquired selectional preferences. In *EMNLP'09*.
- Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *ACL '03*.
- Adam R. Teichert and Hal Daumé III. 2009. Unsupervised part of speech tagging without a lexicon. In *NIPS Workshop on Grammar Induction, Representation of Language and Language Learning*.
- E. Uzun, Y. Klaslan, H.V. Agun, and E. Uar. 2008. Web-based acquisition of subcategorization frames for Turkish. In *The Eighth International Conference on Artificial Intelligence and Soft Computing*.
- Giulia Venturi, Simonetta Montemagni, Simone Marchi, Yutaka Sasaki, Paul Thompson, John McNaught, and Sophia Ananiadou. 2009. Bootstrapping a verb lexicon for biomedical information extraction. In *Computational Linguistics and Intelligent Text Processing*. Springer Berlin / Heidelberg.

# Fast Online Lexicon Learning for Grounded Language Acquisition

David L. Chen

Department of Computer Science  
The University of Texas at Austin  
1616 Guadalupe, Suite 2.408  
Austin, TX 78701, USA  
dlcc@cs.utexas.edu

## Abstract

Learning a semantic lexicon is often an important first step in building a system that learns to interpret the meaning of natural language. It is especially important in language grounding where the training data usually consist of language paired with an ambiguous perceptual context. Recent work by Chen and Mooney (2011) introduced a lexicon learning method that deals with ambiguous relational data by taking intersections of graphs. While the algorithm produced good lexicons for the task of learning to interpret navigation instructions, it only works in batch settings and does not scale well to large datasets. In this paper we introduce a new online algorithm that is an order of magnitude faster and surpasses the state-of-the-art results. We show that by changing the grammar of the formal meaning representation language and training on additional data collected from Amazon's Mechanical Turk we can further improve the results. We also include experimental results on a Chinese translation of the training data to demonstrate the generality of our approach.

## 1 Introduction

Learning to understand the semantics of human languages has been one of the ultimate goals of natural language processing (NLP). Traditional learning approaches have relied on access to parallel corpora of natural language sentences paired with their meanings (Mooney, 2007; Zettlemoyer and Collins, 2007; Lu et al., 2008; Kwiatkowski et al., 2010). However, constructing such semantic annotations can be

difficult and time-consuming. More recently, there has been work on learning from ambiguous supervision where a set of potential sentence meanings are given, only one (or a small subset) of which are correct (Chen and Mooney, 2008; Liang et al., 2009; Bordes et al., 2010; Chen and Mooney, 2011). Given the training data, the system needs to infer the correcting meaning for each training sentence.

Building a lexicon of the formal meaning representations of words and phrases, either implicitly or explicitly, is usually an important step in inferring the meanings of entire sentences. In particular, Chen and Mooney (2011) first learned a lexicon to help them resolve ambiguous supervision of relational data in which the number of choices is exponential. They represent the perceptual context as a graph and allow each sentence in the training data to align to any connected subgraph. Their lexicon learning algorithm finds the common connected subgraph that occurs with a word by taking intersections of the graphs that represent the different contexts in which the word appears. While the algorithm produced a good lexicon for their application of learning to interpret navigation instructions, it only works in batch settings and does not scale well to large datasets. In this paper we introduce a novel online algorithm that is an order of magnitude faster and also produces better results on their navigation task.

In addition to the new lexicon learning algorithm, we also look at modifying the meaning representation grammar (MRG) for their formal semantic language. By using a MRG that correlates better to the structure of natural language, we further improve the performance on the navigation task. Since our al-

gorithm can scale to larger datasets, we present results on collecting and training on additional data from Amazon’s Mechanical Turk. Finally, we show the generality of our approach by demonstrating our system’s ability to learn from a Chinese translation of the training data.

## 2 Background

A common way to learn a lexicon across many different contexts is to find the common parts of the formal representations associated with different occurrences of the same words or phrases (Siskind, 1996). For graphical representations, this involves finding the common subgraph between multiple graphs (Thompson and Mooney, 2003; Chen and Mooney, 2011). In this section we review the lexicon learning algorithm introduced by Chen and Mooney (2011) as well as the overall task they designed to test semantic understanding of navigation instructions.

### 2.1 Navigation Task

The goal of the navigation task is to build a system that can understand free-form natural-language instructions and follow them to move to the intended destination (MacMahon et al., 2006; Shimizu and Haas, 2009; Matuszek et al., 2010; Kollar et al., 2010; Vogel and Jurafsky, 2010; Chen and Mooney, 2011). Chen and Mooney (2011) defined a learning task in which the only supervision the system receives is in the form of observing how humans behave when following sample navigation instructions in a virtual world. Formally, the system is given training data in the form:  $\{(e_1, a_1, w_1), (e_2, a_2, w_2), \dots, (e_n, a_n, w_n)\}$ , where  $e_i$  is a written natural language instruction,  $a_i$  is an observed action sequence, and  $w_i$  is a description of the virtual world. The goal is then to build a system that can produce the correct  $a_j$  given a previously unseen  $(e_j, w_j)$  pair.

Since the observed actions  $a_i$  only consists of low-level actions (e.g. turn left, turn right, walk forward) and not high-level concepts (e.g. turn your back against the wall and walk to the couch), Chen and Mooney first use a set of rules to automatically construct the space of reasonable plans using the action trace and knowledge about the world. The space is represented compactly using a graph as shown in

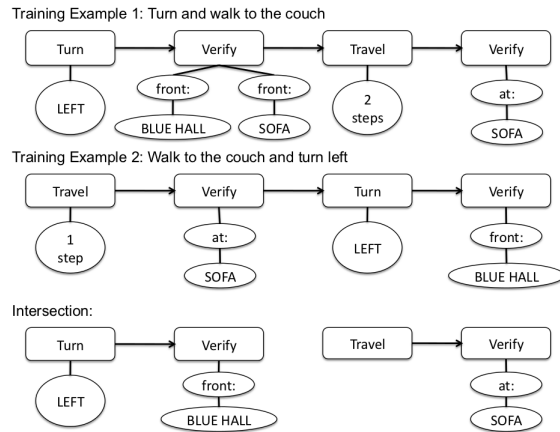


Figure 1: Examples of *landmarks plans* constructed by Chen and Mooney (2011) and how they computed the intersection of two graphs.

Figure 1. This is what they called a *landmarks plan* and consists of the low-level observed actions interleaved with verification steps indicating what objects should be observed after each action.

Given that these *landmarks plans* contain a lot of extraneous details, Chen and Mooney learn a lexicon and use it to identify and remove the irrelevant parts of the plans. They use a greedy method to remove nodes from the graphs that are not associated with any of the words in the instructions. The remaining *refined landmarks plans* are then treated as supervised training data for a semantic-parser learner, KRISP (Kate and Mooney, 2006). Once a semantic parser is trained, it can be used at test time to transform novel instructions into formal navigation plans which are then carried out by a virtual robot (MacMahon et al., 2006).

### 2.2 Lexicon Learning

The central component of the system is the lexicon learning process which associates words and short phrases (n-grams) to their meanings (connected graphs). To learn the meaning of an n-gram  $w$ , Chen and Mooney first collect all navigation plans  $g$  that co-occur with  $w$ . This forms the initial candidate meaning set for  $w$ . They then repeatedly take the intersections between the candidate meanings to generate additional candidate meanings. They use the term intersection to mean a maximal common subgraph (i.e. it is not a subgraph of any other common subgraphs). In general, there are

multiple possible intersections between two graphs. In this case, they bias toward finding large connected components by greedily removing the largest common connected subgraph from both graphs until the two graphs have no overlapping nodes. The output of the intersection process consists of all the removed subgraphs. An example of the intersection operation is shown in Figure 1.

Once the list of candidate meanings are generated, they are ranked by the following scoring metric for an n-gram  $w$  and a graph  $g$ :

$$Score(w, g) = p(g|w) - p(g|\neg w)$$

Intuitively, the score measures how much more likely a graph  $g$  appears when  $w$  is present compared to when it is not. The probabilities are estimated by counting how many examples contain the word  $w$  or graph  $g$ , ignoring multiple occurrences in a single example.

### 3 Online Lexicon Learning Algorithm

While the algorithm presented by Chen and Mooney (2011) produced good lexicons, it only works in batch settings and does not scale well to large datasets. The bottleneck of their algorithm is the intersection process which is time-consuming to perform. Moreover, their algorithm requires taking many intersections between many different graphs. Even though they use beam-search to limit the size of the candidate set, if the initial candidate meaning set for a n-gram is large, it can take a long time to take just one pass through the list of all candidates. Moreover, reducing the beam size could also hurt the quality of the lexicon learned.

In this section, we present another lexicon learning algorithm that is much faster and works in an online setting. The main insight is that most words or short phrases correspond to small graphs. Thus, we concentrate our attention on only candidate meanings that are less than a certain size. Using this constraint, we generate all the potential small connected subgraphs for each navigation plan in the training examples and discard the original graph. Pseudocode for the new algorithm, Subgraph Generation Online Lexicon Learning (SGOLL) algorithm, is shown in Algorithm 1.

As we encounter each new training example which consists of a written navigation instruction

---

#### Algorithm 1 SUBGRAPH GENERATION ONLINE LEXICON LEARNING (SGOLL)

---

**input** A sequence of navigation instructions and the corresponding navigation plans  $(e_1, p_1), \dots, (e_n, p_n)$   
**output** *Lexicon*, a set of phrase-meaning pairs

- 1: **main**
- 2:   **for** training example  $(e_i, p_i)$  **do**
- 3:     Update( $(e_i, p_i)$ )
- 4:   **end for**
- 5:   OutputLexicon()
- 6: **end main**
- 7:
- 8: **function** Update(training example  $(e_i, p_i)$ )
- 9:   **for** n-gram  $w$  that appears in  $e_i$  **do**
- 10:     **for** connected subgraph  $g$  of  $p_i$  such that the size of  $g$  is less than or equal to  $m$  **do**
- 11:       Increase the co-occurrence count of  $g$  and  $w$  by 1
- 12:     **end for**
- 13:   **end for**
- 14:   Increase the count of examples, each n-gram  $w$  and each subgraph  $g$
- 15: **end function**
- 16:
- 17:
- 18: **function** OutputLexicon()
- 19:   **for** n-gram  $w$  that has been observed **do**
- 20:     **if** Number of times  $w$  has been observed is less than *minSup* **then**
- 21:       skip  $w$
- 22:     **end if**
- 23:     **for** subgraph  $g$  that has co-occurred with  $w$  **do**
- 24:       **if** score( $w, g$ ) > threshold  $t$  **then**
- 25:         add  $(w, g)$  to *Lexicon*
- 26:       **end if**
- 27:     **end for**
- 28:   **end for**
- 29: **end function**

---

and the corresponding navigation plan, we first segment the instruction into word tokens and construct n-grams from them. From the corresponding navigation plan, we find all connected subgraphs of size less than or equal to  $m$ . We then update the co-occurrence counts between all the n-grams  $w$  and all the connected subgraphs  $g$ . We also update the counts of how many examples we have encountered so far and counts of the n-grams  $w$  and subgraphs  $g$ . At any given time, we can compute a lexicon using these various counts. Specifically, for each n-gram  $w$ , we look at all the subgraphs  $g$  that co-occurred with it, and compute a score for the pair  $(w, g)$ . If the score is higher than the threshold  $t$ , we add the entry  $(w, g)$  to the lexicon. We use the same scoring function as Chen and Mooney, which can be computed efficiently using the counts we keep. In contrast to Chen and Mooney’s algorithm though, we add the constraint of minimum support by not creating lexical entries for any n-gram  $w$  that appeared in less than  $minSup$  training examples. This is to prevent rarely seen n-grams from receiving high scores in our lexicon simply due to their sparsity. Unless otherwise specified, we compute lexical entries for up to 4-grams with threshold  $t = 0.4$ , maximum subgraph size  $m = 3$ , and minimum support  $minSup = 10$ .

It should be noted that SGOLL can also become computationally intractable if the sizes of the navigations plans are large or if we set the maximum subgraph size  $m$  to a large number. Moreover, the memory requirement can be quite high if there are many different subgraphs  $g$  associated with each n-gram  $w$ . To deal with such scalability issues, we could use beam-search and only keep the top  $k$  candidates associated with each  $w$ . Another important step is to define canonical orderings of the nodes in the graphs. This allows us to determine if two graphs are identical in constant time and also lets us use a hash table to quickly update the co-occurrence and subgraph counts. Thus, even given a large number of subgraphs for each training example, each subgraph can be processed very quickly. Finally, this algorithm readily lends itself to being parallelized. Each processor would get a fraction of the training data and compute the counts individually. Then the counts can be merged together at the end to produce the final lexicon.

### 3.1 Changing the Meaning Representation Grammar

In addition to introducing a new lexicon learning algorithm, we also made another modification to the original system proposed by Chen and Mooney (2011). To train a semantic parser using KRISP (Kate and Mooney, 2006), they had to supply a MRG, a context-free grammar, for their formal navigation plan language. KRISP learns string-kernel classifiers that maps natural language substrings to MRG production rules. Consequently, it is important that the production rules in the MRG mirror the structure of natural language (Kate, 2008).

The original MRG used by Chen and Mooney is a compact grammar that contains many recursive rules that can be used to generate an infinite number of actions or arguments. While these rules are quite expressive, they often do not correspond well to any words or phrases in natural language. To alleviate this problem, we designed another MRG by expanding out many of the rules. For example, the original MRG contained the following production rules for generating an infinite number of travel actions from the root symbol  $S$ .

```
*S -> *Action
*Action -> *Action, *Action
*Action -> *Travel
*Travel -> Travel( )
*Travel -> Travel( steps: *Num )
```

We expand out the production rules as shown below to map  $S$  directly to specific travel actions so they correspond better to patterns such as “go forward” or “walk N steps”.

```
*S -> Travel( )
*S -> Travel( steps: *Num )
*S -> Travel( ), *Action
*S -> Travel( steps: *Num ), *Action
*Action -> *Action, *Action
*Action -> Travel( )
*Action -> Travel( steps: *Num )
```

While this process of expanding the production rules resulted in many more rules, these expanded rules usually correspond better with words or phrases in natural language. We still retain some of the recursive rules to ensure that the formal language remains as expressive as before.

## 4 Collecting Additional Data with Mechanical Turk

One of the motivations for studying ambiguous supervision is the potential ease of acquiring large amounts of training data. Without requiring semantic annotations, a human only has to demonstrate how language is used in context which is generally simple to do. We validate this claim by collecting additional training data for the navigation domain using Mechanical Turk (Snow et al., 2008).

There are two types of data we are interested in collecting: natural language navigation instructions and follower data. Thus, we created two tasks on Mechanical Turk. The first one asks the workers to supply instructions for a randomly generated sequence of actions. The second one asks the workers to try to follow a given navigation instruction in our virtual environment. The latter task is used to generate the corresponding action sequences for instructions collected from the first task.

### 4.1 Task Descriptions

To facilitate the data collection, we first recreated the 3D environments used to collect the original data (MacMahon et al., 2006). We built a Java application that allows the user to freely navigate the three virtual worlds constructed by MacMahon et al. (2006) using the discrete controls of turning left, turning right, and moving forward one step.

The follower task is fairly straightforward using our application. The worker is given a navigation instruction and placed at the starting location. They are asked to follow the navigation instruction as best as they could using the three discrete controls. They could also skip the problem if they did not understand the instruction or if the instruction did not describe a viable route. For each Human Intelligence Task (HIT), we asked the worker to complete 5 follower problems. We paid them \$0.05 for each HIT, or 1 cent per follower problem. The instructions used for the follower problems were mainly collected from our Mechanical Turk instructor task with some of the instructions coming from data collected by MacMahon (2007) that was not used by Chen and Mooney (2011).

The instructor task is slightly more involved because we ask the workers to provide new navigation

	Chen & Mooney	MTurk
# instructions	3236	1011
Vocabulary size	629	590
Avg. # words	7.8 (5.1)	7.69 (7.12)
Avg. # actions	2.1 (2.4)	1.84 (1.24)

Table 1: Statistics about the navigation instruction corpora. The average statistics for each instruction are shown with standard deviations in parentheses.

instructions. The worker is shown a 3D simulation of a randomly generated action sequence between length 1 to 4 and asked to write short, free-form instructions that would lead someone to perform those actions. Since this task requires more time to complete, each HIT consists of only 3 instructor problems. Moreover, we pay the workers \$0.10 for each HIT, or about 3 cents for each instruction they write.

To encourage quality contributions, we use a tiered payment structure (Chen and Dolan, 2011) that rewards the good workers. Workers who have been identified to consistently provide good instructions were allowed to do higher-paying version of the same HITs that paid \$0.15 instead of \$0.10.

### 4.2 Data Statistics

Over a 2-month period we accepted 2,884 follower HITs and 810 instructor HITs from 653 workers. This corresponds to over 14,000 follower traces and 2,400 instructions with most of them consisting of single sentences. For instructions with multiple sentences, we merged all the sentences together and treated it as a single sentence. The total cost of the data collection was \$277.92. While there were 2,400 instructions, we filtered them to make sure they were of reasonable quality. First, we discarded any instructions that did not have at least 5 follower traces. Then we looked at all the follower traces and discarded any instruction that did not have majority agreement on what the correct path is.

Using our strict filter, we were left with slightly over 1,000 instructions. Statistics about the new corpus and the one used by Chen and Mooney can be seen in Table 1. Overall, the new corpus has a slightly smaller vocabulary, and each instruction is slightly shorter both in terms of the number of words and the number of actions.

## 5 Experiments

We evaluate our new lexicon learning algorithm as well as the other modifications to the navigation system using the same three tasks as Chen and Mooney (2011). The first task is disambiguating the training data by inferring the correct navigation plans associated with each training sentence. The second task is evaluating the performance of the semantic parsers trained on the disambiguated data. We measure the performance of both of these tasks by comparing to gold-standard data using the same partial correctness metric used by Chen and Mooney which gives credit to a parse for producing the correct action type and additional credit if the arguments were also correct. Finally, the third task is to complete the end-to-end navigation task. There are two versions of this task, the complete task uses the original instructions which are several sentences long and the other version uses instructions that have been manually split into single sentences. Task completion is measured by the percentage of trials in which the system reached the correct destination (and orientation in the single-sentence version).

We follow the same evaluation scheme as Chen and Mooney and perform leave-one-map-out experiments. For the first task, we build a lexicon using ambiguous training data from two maps, and then use the lexicon to produce the best disambiguated semantic meanings for those same data. For the second and third tasks, we train a semantic parser on the automatically disambiguated data, and test on sentences from the third, unseen map.

For all comparisons to the Chen and Mooney results, we use the performance of their *refined landmarks plans* system which performed the best overall. Moreover, it provides the most direct comparison to our approach since both use a lexicon to refine the *landmarks plans*. Other than the modifications discussed, we use the same components as their system including using KRISP to train the semantic parsers and using the execution module from MacMahon et al. (2006) to carry out the navigation plans.

### 5.1 Inferring Navigation Plans

First, we examine the quality of the refined navigation plans produced using SGOLL’s lexicon. The

	Precision	Recall	F1
Chen and Mooney	78.54	78.10	78.32
SGOLL	87.32	72.96	79.49

Table 2: Partial parse accuracy of how well each algorithm can infer the gold-standard navigation plans.

	Precision	Recall	F1
Chen and Mooney	90.22	55.10	68.37
SGOLL	92.22	55.70	69.43
SGOLL with new MRG	88.36	57.03	69.31

Table 3: Partial parse accuracy of the semantic parsers trained on the disambiguated navigation plans.

precision, recall, and F1 (harmonic mean of precision and recall) of these plans are shown in Table 2. Compared to Chen and Mooney, the plans produced by SGOLL has higher precision and lower recall. This is mainly due to the additional minimum support constraint we added which discards many noisy lexical entries from infrequently seen n-grams.

### 5.2 Training Semantic Parsers

Next we look at the performance of the semantic parsers trained on the inferred navigation plans. The results are shown in Table 3. Here SGOLL performs almost the same as Chen and Mooney, with slightly better precision. We also look at the effect of changing the MRG. Using the new MRG for KRISP to train the semantic parser produced slightly lower precision but higher recall, with similar overall F1 score.

### 5.3 Executing Navigation Plans

Finally, we evaluate the system on the end-to-end navigation task. In addition to SGOLL and SGOLL with the new MRG, we also look at augmenting each of the training splits with the data we collected using Mechanical Turk.

Completion rates for both the single-sentence tasks and the complete tasks are shown in Table 4. Here we see the benefit of each of our modifications. SGOLL outperforms Chen and Mooney’s system on both versions of the navigation task. Using the new MRG to train the semantic parsers further improved performance on both tasks. Finally, augmenting the

	Single-sentence	Complete
Chen and Mooney	54.40%	16.18%
SGOLL	57.09%	17.56%
SGOLL with new MRG	57.28%	19.18%
SGOLL with new MRG and MTurk data	57.62%	20.64%

Table 4: End-to-end navigation task completion rates.

	Computation Time
Chen and Mooney (2011)	2,227.63
SGOLL	157.30
SGOLL with MTurk data	233.27

Table 5: The time (in seconds) it took to build the lexicon.

training data with additional instructions and follower traces collected from Mechanical Turk produced the best results.

#### 5.4 Computation Times

Having established the superior performance of our new system compared to Chen and Mooney’s, we next look at the computational efficiency of SGOLL. The average time (in seconds) it takes for each algorithm to build a lexicon is shown in Table 5. All the results are obtained running the algorithms on Dell PowerEdge 1950 servers with 2x Xeon X5440 (quad-core) 2.83GHz processors and 32GB of RAM. Here SGOLL has a decidedly large advantage over the lexicon learning algorithm from Chen and Mooney, requiring an order of magnitude less time to run. Even after incorporating the new Mechanical Turk data into the training set, SGOLL still takes much less time to build a lexicon. This shows how inefficient it is to perform graph intersection operations and how our online algorithm can more realistically scale to large datasets.

#### 5.5 Experimenting with Chinese Data

In addition to evaluating the system on English data, we also translated the corpus used by Chen and Mooney into Mandarin Chinese.<sup>1</sup> To run our sys-

<sup>1</sup>The translation can be downloaded at <http://www.cs.utexas.edu/~ml/clamp/navigation/>

tem, we first segmented the sentences using the Stanford Chinese Word Segmenter (Chang et al., 2008). We evaluated using the same three tasks as before. This resulted in a precision, recall, and F1 of 87.07, 71.67, and 78.61, respectively for the inferred plans. The trained semantic parser’s precision, recall, and F1 were 88.87, 58.76, and 70.74, respectively. Finally, the system completed 58.70% of the single-sentence task and 20.13% of the complete task. All of these numbers are very similar to the English results, showing the generality of the system in its ability to learn other languages.

#### 5.6 Discussion

We have introduced a novel, online lexicon learning algorithm that is much faster than the one proposed by Chen and Mooney and also performs better on the navigation tasks they devised. Having a computationally efficient algorithm is critical for building systems that learn from ambiguous supervision. Compared to systems that train on supervised semantic annotations, a system that only receives weak, ambiguous training data is expected to have to train on much larger datasets to achieve similar performance. Consequently, such system must be able to scale well in order to keep the learning process tractable. Not only is SGOLL much faster in building a lexicon, it can also be easily parallelized. Moreover, the online nature of SGOLL allows the lexicon to be continually updated while the system is in use. A deployed navigation system can gather new instructions from the user and receive feedback about whether it is performing the correct actions. As new training examples are collected, we can update the corresponding n-gram and subgraph counts without rebuilding the entire lexicon.

One thing to note though is that while SGOLL makes the lexicon learning step much faster and scalable, another bottleneck in the overall system is training the semantic parser. Existing semantic-parser learners such as KRISP were not designed to scale to very large datasets and have trouble training on more than a few thousand examples. Thus, designing new scalable algorithms for learning semantic parsers is critical to scaling the entire system.

We have performed a pilot data collection of new training examples using Mechanical Turk. Even though the instructions were collected from very dif-



ferent sources (paid human subjects from a university for the original data versus workers recruited over the Internet), we showed that adding the new data into the training set improved the system's performance on interpreting instructions from the original corpus. It verified that we are indeed collecting useful information and that non-experts are fully capable of training the system by demonstrating how to use natural language in relevant contexts.

## 6 Related Work

The earliest work on cross-situational word learning was by Siskind (1996) who developed a rule-based system to solve the referential ambiguity problem. However, it did not handle noise and was tested only on artificial data. More recently, Fazly et al. (2010) proposed a probabilistic incremental model that can learn online similar to our algorithm and was tested on transcriptions of child-directed speech. However, they generated the semantic representations from the text itself rather than from the environment. Moreover, the referential ambiguity was introduced artificially by including the correct semantic representation of the neighboring sentence.

Our work falls into the larger framework of learning the semantics of language from weak supervision. This problem can be seen as an alignment problem where each sentence in the training data needs to be aligned to one or more records that represent its meaning. Chen and Mooney (2008) previously introduced another task that aligns sportscasting commentaries to events in a simulated soccer game. Using an EM-like retraining method, they alternated between building a semantic parser and estimating the most likely alignment. Liang et al. (2009) developed an unsupervised approach using a generative model to solve the alignment problem. They demonstrated improved results on matching sentences and events on the sportscasting task and also introduced a new task of aligning weather forecasts to weather information. Kim and Mooney (2010) further improved the generative alignment model by incorporating the full semantic parsing model from Lu et al. (2008). This resulted in a joint generative model that outperformed all previous results. In addition to treating the ambiguous supervision problem as an alignment problem, there

have been other approaches such as treating it as a ranking problem (Bordes et al., 2010), or a PCFG learning problem (Borschinger et al., 2011).

Parallel to the work of learning from ambiguous supervision, other recent work has also looked at training semantic parsers from supervision other than logical-form annotations. Clarke et al. (2010) and Liang et al. (2011) trained systems on question and answer pairs by automatically finding semantic interpretations of the questions that would generate the correct answers. Artzi and Zettlemoyer (2011) use conversation logs between a computer system and a human user to learn to interpret the human utterances. Finally, Goldwasser et al. (2011) presented an unsupervised approach of learning a semantic parser by using an EM-like retraining loop. They use confidence estimation as a proxy for the model's prediction quality, preferring models that have high confidence about their parses.

## 7 Conclusion

Learning the semantics of language from the perceptual context in which it is uttered is a useful approach because only minimal human supervision is required. In this paper we presented a novel online algorithm for building a lexicon from ambiguously supervised relational data. In contrast to the previous approach that computed common subgraphs between different contexts in which an n-gram appeared, we instead focus on small, connected subgraphs and introduce an algorithm, SGOLL, that is an order of magnitude faster. In addition to being more scalable, SGOLL also performed better on the task of interpreting navigation instructions. In addition, we showed that changing the MRG and collecting additional training data from Mechanical Turk further improve the performance of the overall navigation system. Finally, we demonstrated the generality of the system by using it to learn Chinese navigation instructions and achieved similar results.

## Acknowledgments

The research in this paper was supported by the National Science Foundation (NSF) under the grants IIS-0712097 and IIS-1016312. We thank Lu Guo for performing the translation of the corpus into Mandarin Chinese.

## References

- Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*.
- Antoine Bordes, Nicolas Usunier, and Jason Weston. 2010. Label ranking under ambiguous supervision for learning semantic correspondences. In *Proceedings of the 27th International Conference on Machine Learning (ICML-2010)*.
- Benjamin Borschinger, Bevan K. Jones, and Mark Johnson. 2011. Reducing grounded learning tasks to grammatical inference. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*.
- Pi-Chuan Chang, Michel Galley, and Chris Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the ACL Third Workshop on Statistical Machine Translation*.
- David L. Chen and William B. Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011)*, Portland, OR, June.
- David L. Chen and Raymond J. Mooney. 2008. Learning to sportscast: A test of grounded language acquisition. In *Proceedings of 25th International Conference on Machine Learning (ICML-2008)*, Helsinki, Finland, July.
- David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-11)*.
- James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the worlds response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL-2010)*, pages 18–27.
- Afsaneh Fazly, Afra Alishahi, and Suzanne Stevenson. 2010. A probabilistic computational model of cross-situational word learning. *Cognitive Science*, 34(6):1017–1063.
- Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*.
- Rohit J. Kate and Raymond J. Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, pages 913–920, Sydney, Australia, July.
- Rohit J. Kate. 2008. Transforming meaning representation grammars to improve semantic parsing. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, pages 33–40, Manchester, UK, August.
- Joohyun Kim and Raymond J. Mooney. 2010. Generative alignment and semantic parsing for learning from ambiguous supervision. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING-10)*.
- Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. 2010. Toward understanding natural language directions. In *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*.
- Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2010. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP-10)*.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*.
- Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*, Honolulu, HI, October.
- Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: Connecting language, knowledge, and action in route instructions. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06)*.
- Matt MacMahon. 2007. *Following Natural Language Route Instructions*. Ph.D. thesis, Electrical and Computer Engineering Department, University of Texas at Austin.
- Cynthia Matuszek, Dieter Fox, and Karl Koscher. 2010. Following directions using statistical machine translation. In *Proceedings of the 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*.

- Raymond J. Mooney. 2007. Learning for semantic parsing. In A. Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: Proceedings of the 8th International Conference, CICLing 2007, Mexico City*, pages 311–324. Springer Verlag, Berlin.
- Nobuyuki Shimizu and Andrew Haas. 2009. Learning to follow navigational route instructions. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-2009)*.
- Jeffrey M. Siskind. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, 61(1):39–91, October.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast - but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*.
- Cynthia A. Thompson and Raymond J. Mooney. 2003. Acquiring word-meaning mappings for natural language interfaces. *Journal of Artificial Intelligence Research*, 18:1–44.
- Adam Vogel and Dan Jurafsky. 2010. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*.
- Luke S. Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-07)*, pages 678–687, Prague, Czech Republic, June.

# Bayesian Symbol-Refined Tree Substitution Grammars for Syntactic Parsing

Hiroyuki Shindo<sup>†</sup> Yusuke Miyao<sup>‡</sup> Akinori Fujino<sup>†</sup> Masaaki Nagata<sup>†</sup>

<sup>†</sup>NTT Communication Science Laboratories, NTT Corporation  
2-4 Hikaridai, Seika-cho, Soraku-gun, Kyoto, Japan

{shindo.hiroyuki, fujino.akinori, nagata.masaaki}@lab.ntt.co.jp

<sup>‡</sup>National Institute of Informatics  
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan  
yusuke@nii.ac.jp

## Abstract

We propose Symbol-Refined Tree Substitution Grammars (SR-TSGs) for syntactic parsing. An SR-TSG is an extension of the conventional TSG model where each nonterminal symbol can be refined (subcategorized) to fit the training data. We aim to provide a unified model where TSG rules and symbol refinement are learned from training data in a fully automatic and consistent fashion. We present a novel probabilistic SR-TSG model based on the hierarchical Pitman-Yor Process to encode backoff smoothing from a fine-grained SR-TSG to simpler CFG rules, and develop an efficient training method based on Markov Chain Monte Carlo (MCMC) sampling. Our SR-TSG parser achieves an F1 score of 92.4% in the Wall Street Journal (WSJ) English Penn Treebank parsing task, which is a 7.7 point improvement over a conventional Bayesian TSG parser, and better than state-of-the-art discriminative reranking parsers.

## 1 Introduction

Syntactic parsing has played a central role in natural language processing. The resulting syntactic analysis can be used for various applications such as machine translation (Galley et al., 2004; DeNeefe and Knight, 2009), sentence compression (Cohn and Lapata, 2009; Yamangil and Shieber, 2010), and question answering (Wang et al., 2007). Probabilistic context-free grammar (PCFG) underlies many statistical parsers, however, it is well known that the PCFG rules extracted from treebank data via maximum likelihood estimation do not perform well due to unrealistic context freedom assumptions (Klein and Manning, 2003).

In recent years, there has been an increasing interest in tree substitution grammar (TSG) as an alternative to CFG for modeling syntax trees (Post and Gildea, 2009; Tenenbaum et al., 2009; Cohn et al., 2010). TSG is a natural extension of CFG in which nonterminal symbols can be rewritten (substituted) with arbitrarily large tree fragments. These tree fragments have great advantages over tiny CFG rules since they can capture non-local contexts explicitly such as predicate-argument structures, idioms and grammatical agreements (Cohn et al., 2010). Previous work on TSG parsing (Cohn et al., 2010; Post and Gildea, 2009; Bansal and Klein, 2010) has consistently shown that a probabilistic TSG (PTSG) parser is significantly more accurate than a PCFG parser, but is still inferior to state-of-the-art parsers (e.g., the Berkeley parser (Petrov et al., 2006) and the Charniak parser (Charniak and Johnson, 2005)). One major drawback of TSG is that the context freedom assumptions still remain at substitution sites, that is, TSG tree fragments are generated that are conditionally independent of all others given root nonterminal symbols. Furthermore, when a sentence is unparseable with large tree fragments, the PTSG parser usually uses naive CFG rules derived from its backoff model, which diminishes the benefits obtained from large tree fragments.

On the other hand, current state-of-the-art parsers use *symbol refinement* techniques (Johnson, 1998; Collins, 2003; Matsuzaki et al., 2005). Symbol refinement is a successful approach for weakening context freedom assumptions by dividing coarse treebank symbols (e.g. NP and VP) into subcategories, rather than extracting large tree fragments. As shown in several studies on TSG parsing (Zuidema, 2007; Bansal and Klein, 2010), large

tree fragments and symbol refinement work complementarily for syntactic parsing. For example, Bansal and Klein (2010) have reported that deterministic symbol refinement with heuristics helps improve the accuracy of a TSG parser.

In this paper, we propose Symbol-Refined Tree Substitution Grammars (SR-TSGs) for syntactic parsing. SR-TSG is an extension of the conventional TSG model where each nonterminal symbol can be refined (subcategorized) to fit the training data. Our work differs from previous studies in that we focus on a unified model where TSG rules and symbol refinement are learned from training data in a fully automatic and consistent fashion. We also propose a novel probabilistic SR-TSG model with the hierarchical Pitman-Yor Process (Pitman and Yor, 1997), namely a sort of nonparametric Bayesian model, to encode backoff smoothing from a fine-grained SR-TSG to simpler CFG rules, and develop an efficient training method based on blocked MCMC sampling.

Our SR-TSG parser achieves an F1 score of 92.4% in the WSJ English Penn Treebank parsing task, which is a 7.7 point improvement over a conventional Bayesian TSG parser, and superior to state-of-the-art discriminative reranking parsers.

## 2 Background and Related Work

Our SR-TSG work is built upon recent work on Bayesian TSG induction from parse trees (Post and Gildea, 2009; Cohn et al., 2010). We firstly review the Bayesian TSG model used in that work, and then present related work on TSGs and symbol refinement.

A TSG consists of a 4-tuple,  $G = (T, N, S, R)$ , where  $T$  is a set of *terminal symbols*,  $N$  is a set of *nonterminal symbols*,  $S \in N$  is the distinguished *start nonterminal symbol* and  $R$  is a set of productions (a.k.a. rules). The productions take the form of *elementary trees* i.e., tree fragments of height  $\geq 1$ . The root and internal nodes of the elementary trees are labeled with nonterminal symbols, and leaf nodes are labeled with either terminal or nonterminal symbols. Nonterminal leaves are referred to as *frontier nonterminals*, and form the substitution sites to be combined with other elementary trees.

A *derivation* is a process of forming a parse tree. It starts with a root symbol and rewrites (substi-

tutes) nonterminal symbols with elementary trees until there are no remaining frontier nonterminals. Figure 1a shows an example parse tree and Figure 1b shows its example TSG derivation. Since different derivations may produce the same parse tree, recent work on TSG induction (Post and Gildea, 2009; Cohn et al., 2010) employs a probabilistic model of a TSG and predicts derivations from observed parse trees in an unsupervised way.

A Probabilistic Tree Substitution Grammar (PTSG) assigns a probability to each rule in the grammar. The probability of a derivation is defined as the product of the probabilities of its component elementary trees as follows.

$$p(\mathbf{e}) = \prod_{x \rightarrow e \in \mathbf{e}} p(e|x),$$

where  $\mathbf{e} = (e_1, e_2, \dots)$  is a sequence of elementary trees used for the derivation,  $x = \text{root}(e)$  is the root symbol of  $e$ , and  $p(e|x)$  is the probability of generating  $e$  given its root symbol  $x$ . As in a PCFG,  $e$  is generated conditionally independent of all others given  $x$ .

The posterior distribution over elementary trees given a parse tree  $t$  can be computed by using the Bayes' rule:

$$p(\mathbf{e}|t) \propto p(t|\mathbf{e})p(\mathbf{e}).$$

where  $p(t|\mathbf{e})$  is either equal to 1 (when  $t$  and  $\mathbf{e}$  are consistent) or 0 (otherwise). Therefore, the task of TSG induction from parse trees turns out to consist of modeling the prior distribution  $p(\mathbf{e})$ . Recent work on TSG induction defines  $p(\mathbf{e})$  as a nonparametric Bayesian model such as the Dirichlet Process (Ferguson, 1973) or the Pitman-Yor Process to encourage sparse and compact grammars.

Several studies have combined TSG induction and symbol refinement. An adaptor grammar (Johnson et al., 2007a) is a sort of nonparametric Bayesian TSG model with symbol refinement, and is thus closely related to our SR-TSG model. However, an adaptor grammar differs from ours in that all its rules are *complete*: all leaf nodes must be terminal symbols, while our model permits nonterminal symbols as leaf nodes. Furthermore, adaptor grammars have largely been applied to the task of unsupervised structural induction from raw texts such as

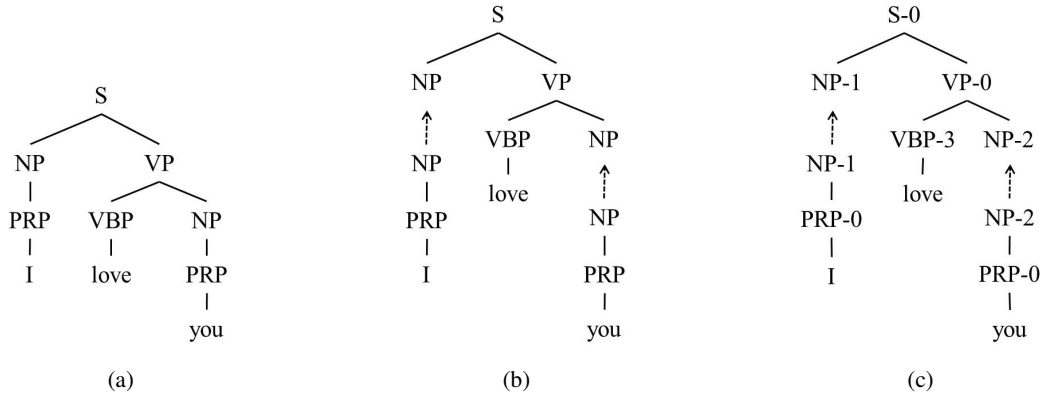


Figure 1: (a) Example parse tree. (b) Example TSG derivation of (a). (c) Example SR-TSG derivation of (a). The refinement annotation is hyphenated with a nonterminal symbol.

morphology analysis, word segmentation (Johnson and Goldwater, 2009), and dependency grammar induction (Cohen et al., 2010), rather than constituent syntax parsing.

An all-fragments grammar (Bansal and Klein, 2010) is another variant of TSG that aims to utilize all possible subtrees as rules. It maps a TSG to an implicit representation to make the grammar tractable and practical for large-scale parsing. The manual symbol refinement described in (Klein and Manning, 2003) was applied to an all-fragments grammar and this improved accuracy in the English WSJ parsing task. As mentioned in the introduction, our model focuses on the automatic learning of a TSG and symbol refinement without heuristics.

### 3 Symbol-Refined Tree Substitution Grammars

In this section, we propose Symbol-Refined Tree Substitution Grammars (SR-TSGs) for syntactic parsing. Our SR-TSG model is an extension of the conventional TSG model where every symbol of the elementary trees can be refined to fit the training data. Figure 1c shows an example of SR-TSG derivation. As with previous work on TSG induction, our task is the induction of SR-TSG derivations from a corpus of parse trees in an unsupervised fashion. That is, we wish to infer the symbol subcategories of every node and substitution site (i.e., nodes where substitution occurs) from parse trees. Extracted rules and their probabilities can be used to parse new raw sentences.

#### 3.1 Probabilistic Model

We define a probabilistic model of an SR-TSG based on the Pitman-Yor Process (PYP) (Pitman and Yor, 1997), namely a sort of nonparametric Bayesian model. The PYP produces power-law distributions, which have been shown to be well-suited for such uses as language modeling (Teh, 2006b), and TSG induction (Cohn et al., 2010). One major issue as regards modeling an SR-TSG is that the space of the grammar rules will be very sparse since SR-TSG allows for arbitrarily large tree fragments and also an arbitrarily large set of symbol subcategories. To address the sparseness problem, we employ a hierarchical PYP to encode a backoff scheme from the SR-TSG rules to simpler CFG rules, inspired by recent work on dependency parsing (Blunsom and Cohn, 2010).

Our model consists of a three-level hierarchy. Table 1 shows an example of the SR-TSG rule and its backoff tree fragments as an illustration of this three-level hierarchy. The topmost level of our model is a distribution over the SR-TSG rules as follows.

$$\begin{aligned}
 e | x_k &\sim G_{x_k} \\
 G_{x_k} &\sim \text{PYP}(d_{x_k}, \theta_{x_k}, P^{\text{sr-tsg}}(\cdot | x_k)),
 \end{aligned}$$

where  $x_k$  is a refined root symbol of an elementary tree  $e$ , while  $x$  is a raw nonterminal symbol in the corpus and  $k = 0, 1, \dots$  is an index of the symbol subcategory. Suppose  $x$  is NP and its symbol subcategory is 0, then  $x_k$  is NP<sub>0</sub>. The PYP has three parameters:  $(d_{x_k}, \theta_{x_k}, P^{\text{sr-tsg}}(\cdot | x_k))$ .

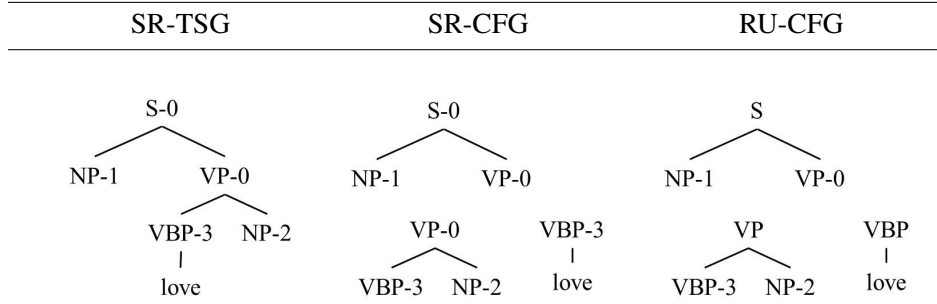


Table 1: Example three-level backoff.

is a *base distribution* over infinite space of symbol-refined elementary trees rooted with  $x_k$ , which provides the backoff probability of  $e$ . The remaining parameters  $d_{x_k}$  and  $\theta_{x_k}$  control the strength of the base distribution.

The backoff probability  $P^{\text{sr-tsg}}(e|x_k)$  is given by the product of *symbol-refined CFG (SR-CFG)* rules that  $e$  contains as follows.

$$\begin{aligned}
 P^{\text{sr-tsg}}(e|x_k) &= \prod_{f \in F(e)} s_{c_f} \times \prod_{i \in I(e)} (1 - s_{c_i}) \\
 &\times H(\text{cfg-rules}(e|x_k)) \\
 \alpha|x_k &\sim H_{x_k} \\
 H_{x_k} &\sim \text{PYP}(d_x, \theta_x, P^{\text{sr-cfg}}(\cdot|x_k)),
 \end{aligned}$$

where  $F(e)$  is a set of frontier nonterminal nodes and  $I(e)$  is a set of internal nodes in  $e$ .  $c_f$  and  $c_i$  are nonterminal symbols of nodes  $f$  and  $i$ , respectively.  $s_c$  is the probability of stopping the expansion of a node labeled with  $c$ . SR-CFG rules are CFG rules where every symbol is refined, as shown in Table 1. The function  $\text{cfg-rules}(e|x_k)$  returns the SR-CFG rules that  $e$  contains, which take the form of  $x_k \rightarrow \alpha$ . Each SR-CFG rule  $\alpha$  rooted with  $x_k$  is drawn from the backoff distribution  $H_{x_k}$ , and  $H_{x_k}$  is produced by the PYP with parameters:  $(d_x, \theta_x, P^{\text{sr-cfg}})$ . This distribution over the SR-CFG rules forms the second level hierarchy of our model.

The backoff probability of the SR-CFG rule,  $P^{\text{sr-cfg}}(\alpha|x_k)$ , is given by the *root-unrefined CFG (RU-CFG)* rule as follows,

$$\begin{aligned}
 P^{\text{sr-cfg}}(\alpha|x_k) &= I(\text{root-unrefine}(\alpha|x_k)) \\
 \alpha|x &\sim I_x \\
 I_x &\sim \text{PYP}(d'_x, \theta'_x, P^{\text{ru-cfg}}(\cdot|x)),
 \end{aligned}$$

where the function  $\text{root-unrefine}(\alpha|x_k)$  returns the RU-CFG rule of  $\alpha$ , which takes the form of  $x \rightarrow \alpha$ . The RU-CFG rule is a CFG rule where the root symbol is unrefined and all leaf nonterminal symbols are refined, as shown in Table 1. Each RU-CFG rule  $\alpha$  rooted with  $x$  is drawn from the backoff distribution  $I_x$ , and  $I_x$  is produced by a PYP. This distribution over the RU-CFG rules forms the third level hierarchy of our model. Finally, we set the backoff probability of the RU-CFG rule,  $P^{\text{ru-cfg}}(\alpha|x)$ , so that it is uniform as follows.

$$P^{\text{ru-cfg}}(\alpha|x) = \frac{1}{|x \rightarrow \cdot|}.$$

where  $|x \rightarrow \cdot|$  is the number of RU-CFG rules rooted with  $x$ . Overall, our hierarchical model encodes backoff smoothing consistently from the SR-TSG rules to the SR-CFG rules, and from the SR-CFG rules to the RU-CFG rules. As shown in (Blunsom and Cohn, 2010; Cohen et al., 2010), the parsing accuracy of the TSG model is strongly affected by its backoff model. The effects of our hierarchical backoff model on parsing performance are evaluated in Section 5.

## 4 Inference

We use Markov Chain Monte Carlo (MCMC) sampling to infer the SR-TSG derivations from parse trees. MCMC sampling is a widely used approach for obtaining random samples from a probability distribution. In our case, we wish to obtain derivation samples of an SR-TSG from the posterior distribution,  $p(\mathbf{e}|\mathbf{t}, \mathbf{d}, \boldsymbol{\theta}, \mathbf{s})$ .

The inference of the SR-TSG derivations corresponds to inferring two kinds of latent variables: latent symbol subcategories and latent substitution

sites. We first infer latent symbol subcategories for every symbol in the parse trees, and then infer latent substitution sites stepwise. During the inference of symbol subcategories, every internal node is fixed as a substitution site. After that, we unfix that assumption and infer latent substitution sites given symbol-refined parse trees. This stepwise learning is simple and efficient in practice, but we believe that the joint learning of both latent variables is possible, and we will deal with this in future work. Here we describe each inference algorithm in detail.

#### 4.1 Inference of Symbol Subcategories

For the inference of latent symbol subcategories, we adopt split and merge training (Petrov et al., 2006) as follows. In each split-merge step, each symbol is split into at most two subcategories. For example, every NP symbol in the training data is split into either  $NP_0$  or  $NP_1$  to maximize the posterior probability. After convergence, we measure the loss of each split symbol in terms of the likelihood incurred when removing it, then the smallest 50% of the newly split symbols as regards that loss are merged to avoid overfitting. The split-merge algorithm terminates when the total number of steps reaches the user-specified value.

In each splitting step, we use two types of blocked MCMC algorithm: the *sentence-level* blocked Metropolis-Hastings (MH) sampler and the *tree-level* blocked Gibbs sampler, while (Petrov et al., 2006) use a different MLE-based model and the EM algorithm. Our sampler iterates sentence-level sampling and tree-level sampling alternately.

The sentence-level MH sampler is a recently proposed algorithm for grammar induction (Johnson et al., 2007b; Cohn et al., 2010). In this work, we apply it to the training of symbol splitting. The MH sampler consists of the following three steps: for each sentence, 1) calculate the inside probability (Lari and Young, 1991) in a bottom-up manner, 2) sample a derivation tree in a top-down manner, and 3) accept or reject the derivation sample by using the MH test. See (Cohn et al., 2010) for details. This sampler simultaneously updates blocks of latent variables associated with a sentence, thus it can find MAP solutions efficiently.

The tree-level blocked Gibbs sampler focuses on the type of SR-TSG rules and simultaneously up-

dates all root and child nodes that are annotated with the same SR-TSG rule. For example, the sampler collects all nodes that are annotated with  $S_0 \rightarrow NP_1VP_2$ , then updates those nodes to another subcategory such as  $S_0 \rightarrow NP_2VP_0$  according to the posterior distribution. This sampler is similar to table label resampling (Johnson and Goldwater, 2009), but differs in that our sampler can update multiple table labels simultaneously when multiple tables are labeled with the same elementary tree. The tree-level sampler also simultaneously updates blocks of latent variables associated with the type of SR-TSG rules, thus it can find MAP solutions efficiently.

#### 4.2 Inference of Substitution Sites

After the inference of symbol subcategories, we use Gibbs sampling to infer the substitution sites of parse trees as described in (Cohn and Lapata, 2009; Post and Gildea, 2009). We assign a binary variable to each internal node in the training data, which indicates whether that node is a substitution site or not. For each iteration, the Gibbs sampler works by sampling the value of each binary variable in random order. See (Cohn et al., 2010) for details.

During the inference, our sampler ignores the symbol subcategories of internal nodes of elementary trees since they do not affect the derivation of the SR-TSG. For example, the elementary trees “ $(S_0 (NP_0 NNP_0) VP_0)$ ” and “ $(S_0 (NP_1 NNP_0) VP_0)$ ” are regarded as being the same when we calculate the generation probabilities according to our model. This heuristic is helpful for finding large tree fragments and learning compact grammars.

#### 4.3 Hyperparameter Estimation

We treat hyperparameters  $\{\mathbf{d}, \boldsymbol{\theta}\}$  as random variables and update their values for every MCMC iteration. We place a prior on the hyperparameters as follows:  $d \sim \text{Beta}(1, 1)$ ,  $\theta \sim \text{Gamma}(1, 1)$ . The values of  $d$  and  $\theta$  are optimized with the auxiliary variable technique (Teh, 2006a).



## 5 Experiment

### 5.1 Settings

#### 5.1.1 Data Preparation

We ran experiments on the Wall Street Journal (WSJ) portion of the English Penn Treebank data set (Marcus et al., 1993), using a standard data split (sections 2–21 for training, 22 for development and 23 for testing). We also used section 2 as a *small* training set for evaluating the performance of our model under low-resource conditions. Henceforth, we distinguish the small training set (section 2) from the full training set (sections 2-21). The treebank data is right-binarized (Matsuzaki et al., 2005) to construct grammars with only unary and binary productions. We replace lexical words with count  $\leq 5$  in the training data with one of 50 unknown words using lexical features, following (Petrov et al., 2006). We also split off all the function tags and eliminated empty nodes from the data set, following (Johnson, 1998).

#### 5.1.2 Training and Parsing

For the inference of symbol subcategories, we trained our model with the MCMC sampler by using 6 split-merge steps for the full training set and 3 split-merge steps for the small training set. Therefore, each symbol can be subdivided into a maximum of  $2^6 = 64$  and  $2^3 = 8$  subcategories, respectively. In each split-merge step, we initialized the sampler by randomly splitting every symbol in two subcategories and ran the MCMC sampler for 1000 iterations. After that, to infer the substitution sites, we initialized the model with the final sample from a run on the small training set, and used the Gibbs sampler for 2000 iterations. We estimated the optimal values of the stopping probabilities  $s$  by using the development set.

We obtained the parsing results with the MAX-RULE-PRODUCT algorithm (Petrov et al., 2006) by using the SR-TSG rules extracted from our model. We evaluated the accuracy of our parser by bracketing F1 score of predicted parse trees. We used EVALB<sup>1</sup> to compute the F1 score. In all our experiments, we conducted ten independent runs to train our model, and selected the one that performed best on the development set in terms of parsing accuracy.

<sup>1</sup><http://nlp.cs.nyu.edu/evalb/>

Model	F1 (small)	F1 (full)
CFG	61.9	63.6
*TSG	77.1	85.0
SR-TSG ( $P^{sr-tsg}$ )	73.0	86.4
SR-TSG ( $P^{sr-tsg}, P^{sr-cfg}$ )	79.4	89.7
SR-TSG ( $P^{sr-tsg}, P^{sr-cfg}, P^{ru-cfg}$ )	<b>81.7</b>	<b>91.1</b>

Table 2: Comparison of parsing accuracy with the small and full training sets. \*Our reimplementation of (Cohn et al., 2010).

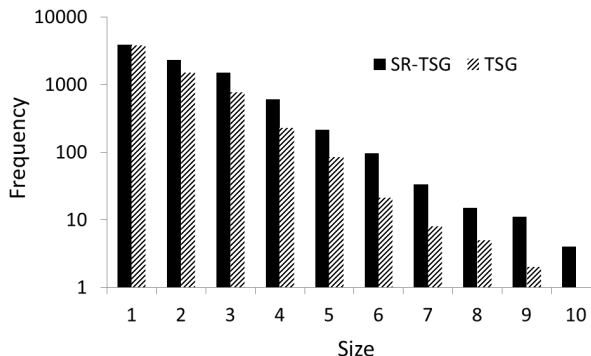


Figure 2: Histogram of SR-TSG and TSG rule sizes on the small training set. The size is defined as the number of CFG rules that the elementary tree contains.

## 5.2 Results and Discussion

### 5.2.1 Comparison of SR-TSG with TSG

We compared the SR-TSG model with the CFG and TSG models as regards parsing accuracy. We also tested our model with three backoff hierarchy settings to evaluate the effects of backoff smoothing on parsing accuracy. Table 2 shows the F1 scores of the CFG, TSG and SR-TSG parsers for small and full training sets. In Table 2, SR-TSG ( $P^{sr-tsg}$ ) denotes that we used only the topmost level of the hierarchy. Similarly, SR-TSG ( $P^{sr-tsg}, P^{sr-cfg}$ ) denotes that we used only the  $P^{sr-tsg}$  and  $P^{sr-cfg}$  backoff models.

Our best model, SR-TSG ( $P^{sr-tsg}, P^{sr-cfg}, P^{ru-cfg}$ ), outperformed both the CFG and TSG models on both the small and large training sets. This result suggests that the conventional TSG model trained from the vanilla treebank is insufficient to resolve

Model	F1 ( $\leq 40$ )	F1 (all)
TSG (no symbol refinement)		
Post and Gildea (2009)	82.6	-
Cohn et al. (2010)	85.4	84.7
TSG with Symbol Refinement		
Zuidema (2007)	-	*83.8
Bansal et al. (2010)	88.7	88.1
<b>SR-TSG (single)</b>	<b>91.6</b>	<b>91.1</b>
<b>SR-TSG (multiple)</b>	<b>92.9</b>	<b>92.4</b>
CFG with Symbol Refinement		
Collins (1999)	88.6	88.2
Petrov and Klein (2007)	90.6	90.1
Petrov (2010)	-	91.8
Discriminative		
Carreras et al. (2008)	-	91.1
Charniak and Johnson (2005)	92.0	91.4
Huang (2008)	92.3	91.7

Table 3: Our parsing performance for the testing set compared with those of other parsers. \*Results for the development set ( $\leq 100$ ).

structural ambiguities caused by coarse symbol annotations in a training corpus. As we expected, symbol refinement can be helpful with the TSG model for further fitting the training set and improving the parsing accuracy.

The performance of the SR-TSG parser was strongly affected by its backoff models. For example, the simplest model,  $P^{SR-TSG}$ , performed poorly compared with our best model. This result suggests that the SR-TSG rules extracted from the training set are very sparse and cannot cover the space of unknown syntax patterns in the testing set. Therefore, sophisticated backoff modeling is essential for the SR-TSG parser. Our hierarchical PYP modeling technique is a successful way to achieve backoff smoothing from sparse SR-TSG rules to simpler CFG rules, and offers the advantage of automatically estimating the optimal backoff probabilities from the training set.

We compared the rule sizes and frequencies of SR-TSG with those of TSG. The rule sizes of SR-

TSG and TSG are defined as the number of CFG rules that the elementary tree contains. Figure 2 shows a histogram of the SR-TSG and TSG rule sizes (by *unrefined* token) on the small training set. For example, SR-TSG rules:  $S_1 \rightarrow NP_0VP_1$  and  $S_0 \rightarrow NP_1VP_2$  were considered to be the same token. In Figure 2, we can see that there are almost the same number of SR-TSG rules and TSG rules with size = 1. However, there are more SR-TSG rules than TSG rules with size  $\geq 2$ . This shows that an SR-TSG can use various large tree fragments depending on the context, which is specified by the symbol subcategories.

### 5.2.2 Comparison of SR-TSG with Other Models

We compared the accuracy of the SR-TSG parser with that of conventional high-performance parsers. Table 3 shows the F1 scores of an SR-TSG and conventional parsers with the full training set. In Table 3, SR-TSG (single) is a standard SR-TSG parser,

and SR-TSG (multiple) is a combination of sixteen independently trained SR-TSG models, following the work of (Petrov, 2010).

Our SR-TSG (single) parser achieved an F1 score of 91.1%, which is a 6.4 point improvement over the conventional Bayesian TSG parser reported by (Cohn et al., 2010). Our model can be viewed as an extension of Cohn’s work by the incorporation of symbol refinement. Therefore, this result confirms that a TSG and symbol refinement work complementarily in improving parsing accuracy. Compared with a symbol-refined CFG model such as the Berkeley parser (Petrov et al., 2006), the SR-TSG model can use large tree fragments, which strengthens the probability of frequent syntax patterns in the training set. Indeed, the few very large rules of our model memorized full parse trees of sentences, which were repeated in the training set.

The SR-TSG (single) is a pure generative model of syntax trees but it achieved results comparable to those of discriminative parsers. It should be noted that discriminative reranking parsers such as (Charniak and Johnson, 2005) and (Huang, 2008) are constructed on a generative parser. The reranking parser takes the k-best lists of candidate trees or a packed forest produced by a baseline parser (usually a generative model), and then reranks the candidates using arbitrary features. Hence, we can expect that combining our SR-TSG model with a discriminative reranking parser would provide better performance than SR-TSG alone.

Recently, (Petrov, 2010) has reported that combining multiple grammars trained independently gives significantly improved performance over a single grammar alone. We applied his method (referred to as a TREE-LEVEL inference) to the SR-TSG model as follows. We first trained sixteen SR-TSG models independently and produced a 100-best list of the derivations for each model. Then, we erased the subcategory information of parse trees and selected the best tree that achieved the highest likelihood under the product of sixteen models. The combination model, SR-TSG (multiple), achieved an F1 score of 92.4%, which is a state-of-the-art result for the WSJ parsing task. Compared with discriminative reranking parsers, combining multiple grammars by using the product model provides the advantage that it does not require any additional training. Several

studies (Fossum and Knight, 2009; Zhang et al., 2009) have proposed different approaches that involve combining k-best lists of candidate trees. We will deal with those methods in future work.

Let us note the relation between SR-CFG, TSG and SR-TSG. TSG is weakly equivalent to CFG and generates the same set of strings. For example, the TSG rule “ $S \rightarrow (NP\ NNP)\ VP$ ” with probability  $p$  can be converted to the equivalent CFG rules as follows: “ $S \rightarrow NP^{NNP}\ VP$ ” with probability  $p$  and “ $NP^{NNP} \rightarrow NNP$ ” with probability 1. From this viewpoint, TSG utilizes surrounding symbols (NNP of  $NP^{NNP}$  in the above example) as latent variables with which to capture context information. The search space of learning a TSG given a parse tree is  $\mathcal{O}(2^n)$  where  $n$  is the number of internal nodes of the parse tree. On the other hand, an SR-CFG utilizes an arbitrary index such as  $0, 1, \dots$  as latent variables and the search space is larger than that of a TSG when the symbol refinement model allows for more than two subcategories for each symbol. Our experimental results confirm that jointly modeling both latent variables using our SR-TSG assists accurate parsing.

## 6 Conclusion

We have presented an SR-TSG, which is an extension of the conventional TSG model where each symbol of tree fragments can be automatically subcategorized to address the problem of the conditional independence assumptions of a TSG. We proposed a novel backoff modeling of an SR-TSG based on the hierarchical Pitman-Yor Process and sentence-level and tree-level blocked MCMC sampling for training our model. Our best model significantly outperformed the conventional TSG and achieved state-of-the-art result in a WSJ parsing task. Future work will involve examining the SR-TSG model for different languages and for unsupervised grammar induction.

## Acknowledgements

We would like to thank Liang Huang for helpful comments and the three anonymous reviewers for thoughtful suggestions. We would also like to thank Slav Petrov and Hui Zhang for answering our questions about their parsers.

## References

- Mohit Bansal and Dan Klein. 2010. Simple, Accurate Parsing with an All-Fragments Grammar. In *In Proc. of ACL*, pages 1098–1107.
- Phil Blunsom and Trevor Cohn. 2010. Unsupervised Induction of Tree Substitution Grammars for Dependency Parsing. In *Proc. of EMNLP*, pages 1204–1213.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proc. of ACL*, 1:173–180.
- Shay B Cohen, David M Blei, and Noah A Smith. 2010. Variational Inference for Adaptor Grammars. In *In Proc. of HLT-NAACL*, pages 564–572.
- Trevor Cohn and Mirella Lapata. 2009. Sentence Compression as Tree Transduction. *Journal of Artificial Intelligence Research*, 34:637–674.
- Trevor Cohn, Phil Blunsom, and Sharon Goldwater. 2010. Inducing Tree-Substitution Grammars. *Journal of Machine Learning Research*, 11:3053–3096.
- Michael Collins. 2003. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics*, 29:589–637.
- Steve DeNeefe and Kevin Knight. 2009. Synchronous Tree Adjoining Machine Translation. In *Proc. of EMNLP*, page 727.
- Thomas S Ferguson. 1973. A Bayesian Analysis of Some Nonparametric Problems. *Annals of Statistics*, 1:209–230.
- Victoria Fossum and Kevin Knight. 2009. Combining Constituent Parsers. In *Proc. of HLT-NAACL*, pages 253–256.
- Michel Galley, Mark Hopkins, Kevin Knight, Daniel Marcu, Los Angeles, and Marina Del Rey. 2004. What’s in a Translation Rule? *Information Sciences*, pages 273–280.
- Liang Huang. 2008. Forest Reranking : Discriminative Parsing with Non-Local Features. In *Proc. of ACL*, 19104:0.
- Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *In Proc. of HLT-NAACL*, pages 317–325.
- Mark Johnson, Thomas L Griffiths, and Sharon Goldwater. 2007a. Adaptor Grammars : A Framework for Specifying Compositional Nonparametric Bayesian Models. *Advances in Neural Information Processing Systems 19*, 19:641–648.
- Mark Johnson, Thomas L Griffiths, and Sharon Goldwater. 2007b. Bayesian Inference for PCFGs via Markov chain Monte Carlo. In *In Proc. of HLT-NAACL*, pages 139–146.
- Mark Johnson. 1998. PCFG Models of Linguistic Tree Representations. *Computational Linguistics*, 24:613–632.
- Dan Klein and Christopher D Manning. 2003. Accurate Unlexicalized Parsing. In *Proc. of ACL*, 1:423–430.
- K Lari and S J Young. 1991. Applications of Stochastic Context-Free Grammars Using the Inside–Outside Algorithm. *Computer Speech and Language*, 5:237–257.
- Mitchell P Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.
- Takuya Matsuzaki, Yusuke Miyao, and Jun’ichi Tsujii. 2005. Probabilistic CFG with latent annotations. In *Proc. of ACL*, pages 75–82.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proc. of ACL*, pages 433–440.
- Slav Petrov. 2010. Products of Random Latent Variable Grammars. In *Proc. of HLT-NAACL*, pages 19–27.
- Jim Pitman and Marc Yor. 1997. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25:855–900.
- Matt Post and Daniel Gildea. 2009. Bayesian Learning of a Tree Substitution Grammar. In *In Proc. of ACL-IJCNLP*, pages 45–48.
- Yee Whye Teh. 2006a. A Bayesian Interpretation of Interpolated Kneser-Ney. *NUS School of Computing Technical Report TRA2/06*.
- YW Teh. 2006b. A Hierarchical Bayesian Language Model based on Pitman-Yor Processes. In *Proc. of ACL*, 44:985–992.
- J Tenenbaum, TJ O’Donnell, and ND Goodman. 2009. Fragment Grammars: Exploring Computation and Reuse in Language. *MIT Computer Science and Artificial Intelligence Laboratory Technical Report Series*.
- Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the Jeopardy Model ? A Quasi-Synchronous Grammar for QA. In *Proc. of EMNLP-CoNLL*, pages 22–32.
- Elif Yamangil and Stuart M Shieber. 2010. Bayesian Synchronous Tree-Substitution Grammar Induction and Its Application to Sentence Compression. In *In Proc. of ACL*, pages 937–947.
- Hui Zhang, Min Zhang, Chew Lim Tan, and Haizhou Li. 2009. K-Best Combination of Syntactic Parsers. In *Proc. of EMNLP*, pages 1552–1560.
- Willem Zuidema. 2007. Parsimonious Data-Oriented Parsing. In *Proc. of EMNLP-CoNLL*, pages 551–560.

# String Re-writing Kernel

Fan Bu<sup>1</sup>, Hang Li<sup>2</sup> and Xiaoyan Zhu<sup>3</sup>

<sup>1,3</sup>State Key Laboratory of Intelligent Technology and Systems

<sup>1,3</sup>Tsinghua National Laboratory for Information Sci. and Tech.

<sup>1,3</sup>Department of Computer Sci. and Tech., Tsinghua University

<sup>2</sup>Microsoft Research Asia, No. 5 Danling Street, Beijing 100080, China

<sup>1</sup>bufan0000@gmail.com

<sup>2</sup>hangli@microsoft.com

<sup>3</sup>zxy-dcs@tsinghua.edu.cn

## Abstract

Learning for sentence re-writing is a fundamental task in natural language processing and information retrieval. In this paper, we propose a new class of kernel functions, referred to as string re-writing kernel, to address the problem. A string re-writing kernel measures the similarity between *two pairs of strings*, each pair representing re-writing of a string. It can capture the lexical and structural similarity between two pairs of sentences without the need of constructing syntactic trees. We further propose an instance of string re-writing kernel which can be computed efficiently. Experimental results on benchmark datasets show that our method can achieve better results than state-of-the-art methods on two sentence re-writing learning tasks: paraphrase identification and recognizing textual entailment.

## 1 Introduction

Learning for sentence re-writing is a fundamental task in natural language processing and information retrieval, which includes paraphrasing, textual entailment and transformation between query and document title in search.

The key question here is how to represent the re-writing of sentences. In previous research on sentence re-writing learning such as paraphrase identification and recognizing textual entailment, most representations are based on the lexicons (Zhang and Patrick, 2005; Lintean and Rus, 2011; de Marneffe et al., 2006) or the syntactic trees (Das and Smith,

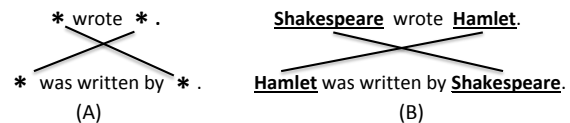


Figure 1: Example of re-writing. (A) is a re-writing rule and (B) is a re-writing of sentence.

2009; Heilman and Smith, 2010) of the sentence pairs.

In (Lin and Pantel, 2001; Barzilay and Lee, 2003), re-writing rules serve as underlying representations for paraphrase generation/discovery. Motivated by the work, we represent re-writing of sentences by all possible re-writing rules that can be applied into it. For example, in Fig. 1, (A) is one re-writing rule that can be applied into the sentence re-writing (B). Specifically, we propose a new class of kernel functions (Schölkopf and Smola, 2002), called string re-writing kernel (SRK), which defines the similarity between two re-writings (pairs) of strings as the inner product between them in the feature space induced by all the re-writing rules. SRK is different from existing kernels in that it is for re-writing and defined on *two pairs of strings*. SRK can capture the lexical and structural similarity between re-writings of sentences and does not need to parse the sentences and create the syntactic trees of them.

One challenge for using SRK lies in the high computational cost of straightforwardly computing the kernel, because it involves two re-writings of strings (i.e., four strings) and a large number of re-writing rules. We are able to develop an instance of SRK, referred to as kb-SRK, which directly computes the number of common rewriting rules without explic-

itly calculating the inner product between feature vectors, and thus drastically reduce the time complexity.

Experimental results on benchmark datasets show that SRK achieves better results than the state-of-the-art methods in paraphrase identification and recognizing textual entailment. Note that SRK is very flexible to the formulations of sentences. For example, informally written sentences such as long queries in search can also be effectively handled.

## 2 Related Work

The string kernel function, first proposed by Lodhi et al. (2002), measures the similarity between two strings by their shared substrings. Leslie et al. (2002) proposed the  $k$ -spectrum kernel which represents strings by their contiguous substrings of length  $k$ . Leslie et al. (2004) further proposed a number of string kernels including the wildcard kernel to facilitate inexact matching between the strings. The string kernels defined on two pairs of objects (including strings) were also developed, which decompose the similarity into product of similarities between individual objects using tensor product (Basilico and Hofmann, 2004; Ben-Hur and Noble, 2005) or Cartesian product (Kashima et al., 2009).

The task of paraphrasing usually consists of paraphrase pattern generation and paraphrase identification. Paraphrase pattern generation is to automatically extract semantically equivalent patterns (Lin and Pantel, 2001; Bhagat and Ravichandran, 2008) or sentences (Barzilay and Lee, 2003). Paraphrase identification is to identify whether two given sentences are a paraphrase of each other. The methods proposed so far formalized the problem as classification and used various types of features such as bag-of-words feature, edit distance (Zhang and Patrick, 2005), dissimilarity kernel (Linteau and Rus, 2011) predicate-argument structure (Qiu et al., 2006), and tree edit model (which is based on a tree kernel) (Heilman and Smith, 2010) in the classification task. Among the most successful methods, Wan et al. (2006) enriched the feature set by the BLEU metric and dependency relations. Das and Smith (2009) used the quasi-synchronous grammar formalism to incorporate features from WordNet, named entity recognizer, POS tagger, and dependency la-

bels from aligned trees.

The task of recognizing textual entailment is to decide whether the hypothesis sentence can be entailed by the premise sentence (Giampiccolo et al., 2007). In recognizing textual entailment, de Marneffe et al. (2006) classified sentences pairs on the basis of word alignments. MacCartney and Manning (2008) used an inference procedure based on natural logic and combined it with the methods by de Marneffe et al. (2006). Harmeling (2007) and Heilman and Smith (2010) classified sequence pairs based on transformation on syntactic trees. Zanzotto et al. (2007) used a kernel method on syntactic tree pairs (Moschitti and Zanzotto, 2007).

## 3 Kernel Approach to Sentence Re-Writing Learning

We formalize sentence re-writing learning as a kernel method. Following the literature of string kernel, we use the terms “string” and “character” instead of “sentence” and “word”.

Suppose that we are given training data consisting of re-writings of strings and their responses

$$((s_1, t_1), y_1), \dots, ((s_n, t_n), y_n) \in (\Sigma^* \times \Sigma^*) \times Y$$

where  $\Sigma$  denotes the character set,  $\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$  denotes the string set, which is the Kleene closure of set  $\Sigma$ ,  $Y$  denotes the set of responses, and  $n$  is the number of instances.  $(s_i, t_i)$  is a re-writing consisting of the source string  $s_i$  and the target string  $t_i$ .  $y_i$  is the response which can be a category, ordinal number, or real number. In this paper, for simplicity we assume that  $Y = \{\pm 1\}$  (e.g. paraphrase/non-paraphrase). Given a new string re-writing  $(s, t) \in \Sigma^* \times \Sigma^*$ , our goal is to predict its response  $y$ . That is, the training data consists of binary classes of string re-writings, and the prediction is made for the new re-writing based on learning from the training data.

We take the kernel approach to address the learning task. The kernel on re-writings of strings is defined as

$$K : (\Sigma^* \times \Sigma^*) \times (\Sigma^* \times \Sigma^*) \rightarrow \mathbb{R}$$

satisfying for all  $(s_i, t_i), (s_j, t_j) \in \Sigma^* \times \Sigma^*$ ,

$$K((s_i, t_i), (s_j, t_j)) = \langle \Phi(s_i, t_i), \Phi(s_j, t_j) \rangle$$

where  $\Phi$  maps each re-writing (pair) of strings into a high dimensional Hilbert space  $\mathcal{H}$ , referred to as

feature space. By the representer theorem (Kimeldorf and Wahba, 1971; Schölkopf and Smola, 2002), it can be shown that the response  $y$  of a new string re-writing  $(s, t)$  can always be represented as

$$y = \text{sign}\left(\sum_{i=1}^n \alpha_i y_i K((s_i, t_i), (s, t))\right)$$

where  $\alpha_i \geq 0$ ,  $(i = 1, \dots, n)$  are parameters. That is, it is determined by a linear combination of the similarities between the new instance and the instances in training set. It is also known that by employing a learning model such as SVM (Vapnik, 2000), such a linear combination can be automatically learned by solving a quadratic optimization problem. The question then becomes how to design the kernel function for the task.

#### 4 String Re-writing Kernel

Let  $\Sigma$  be the set of characters and  $\Sigma^*$  be the set of strings. Let wildcard domain  $D \subseteq \Sigma^*$  be the set of strings which can be replaced by wildcards.

The string re-writing kernel measures the similarity between two string re-writings through the re-writing rules that can be applied into them. Formally, given re-writing rule set  $R$  and wildcard domain  $D$ , the *string re-writing kernel (SRK)* is defined as

$$K((s_1, t_1), (s_2, t_2)) = \langle \Phi(s_1, t_1), \Phi(s_2, t_2) \rangle \quad (1)$$

where  $\Phi(s, t) = (\phi_r(s, t))_{r \in R}$  and

$$\phi_r(s, t) = n\lambda^i \quad (2)$$

where  $n$  is the number of contiguous substring pairs of  $(s, t)$  that re-writing rule  $r$  matches,  $i$  is the number of wildcards in  $r$ , and  $\lambda \in (0, 1]$  is a factor punishing each occurrence of wildcard.

A *re-writing rule* is defined as a triple  $r = (\beta_s, \beta_t, \tau)$  where  $\beta_s, \beta_t \in (\Sigma \cup \{*\})^*$  denote source and target string patterns and  $\tau \subseteq \text{ind}_*(\beta_s) \times \text{ind}_*(\beta_t)$  denotes the alignments between the wildcards in the two string patterns. Here  $\text{ind}_*(\beta)$  denotes the set of indexes of wildcards in  $\beta$ .

We say that a re-writing rule  $(\beta_s, \beta_t, \tau)$  *matches* a string pair  $(s, t)$ , if and only if string patterns  $\beta_s$  and  $\beta_t$  can be changed into  $s$  and  $t$  respectively by substituting each wildcard in the string patterns with an element in the strings, where the elements are defined in the wildcard domain  $D$  and the wildcards

$\beta_s[i]$  and  $\beta_t[j]$  are substituted by the same elements, when there is an alignment  $(i, j) \in \tau$ .

For example, the re-writing rule in Fig. 1 (A) can be formally written as  $r = (\beta_s, \beta_t, \tau)$  where  $\beta_s = (*, \text{wrote}, *)$ ,  $\beta_t = (*, \text{was}, \text{written}, \text{by}, *)$  and  $\tau = \{(1, 5), (3, 1)\}$ . It matches with the string pair in Fig. 1 (B).

String re-writing kernel is a class of kernels which depends on re-writing rule set  $R$  and wildcard domain  $D$ . Here we provide some examples. Obviously, the effectiveness and efficiency of SRK depend on the choice of  $R$  and  $D$ .

**Example 1.** We define the pairwise  $k$ -spectrum kernel (*ps-SRK*)  $K_k^{ps}$  as the re-writing rule kernel under  $R = \{(\beta_s, \beta_t, \tau) \mid \beta_s, \beta_t \in \Sigma^k, \tau = \emptyset\}$  and any  $D$ . It can be shown that  $K_k^{ps}((s_1, t_1), (s_2, t_2)) = K_k^{spec}(s_1, s_2)K_k^{spec}(t_1, t_2)$  where  $K_k^{spec}(x, y)$  is equivalent to the  $k$ -spectrum kernel proposed by Leslie et al. (2002).

**Example 2.** The pairwise  $k$ -wildcard kernel (*pw-SRK*)  $K_k^{pw}$  is defined as the re-writing rule kernel under  $R = \{(\beta_s, \beta_t, \tau) \mid \beta_s, \beta_t \in (\Sigma \cup \{*\})^k, \tau = \emptyset\}$  and  $D = \Sigma$ . It can be shown that  $K_k^{pw}((s_1, t_1), (s_2, t_2)) = K_{(k,k)}^{wc}(s_1, s_2)K_{(k,k)}^{wc}(t_1, t_2)$  where  $K_{(k,k)}^{wc}(x, y)$  is a special case ( $m=k$ ) of the  $(k, m)$ -wildcard kernel proposed by Leslie et al. (2004).

Both kernels shown above are represented as the product of two kernels defined separately on strings  $s_1, s_2$  and  $t_1, t_2$ , and that is to say that they do not consider the alignment relations between the strings.

#### 5 K-gram Bijective String Re-writing Kernel

Next we propose another instance of string re-writing kernel, called the  $k$ -gram bijective string re-writing kernel (*kb-SRK*). As will be seen, kb-SRK can be computed efficiently, although it is defined on two pairs of strings and is not decomposed (note that ps-SRK and pw-SRK are decomposed).

##### 5.1 Definition

The kb-SRK has the following properties: (1) A wildcard can only substitute a single character, denoted as “?”. (2) The two string patterns in a re-writing rule are of length  $k$ . (3) The alignment relation in a re-writing rule is bijective, i.e., there is a one-to-one mapping between the wildcards in

the string patterns. Formally, the  $k$ -gram *bijective string re-writing kernel*  $\mathbb{K}_k$  is defined as a string re-writing kernel under the re-writing rule set  $\mathbf{R} = \{(\beta_s, \beta_t, \tau) \mid \beta_s, \beta_t \in (\Sigma \cup \{?\})^k, \tau \text{ is bijective}\}$  and the wildcard domain  $\mathbf{D} = \Sigma$ .

Since each re-writing rule contains two string patterns of length  $k$  and each wildcard can only substitute one character, a re-writing rule can only match  $k$ -gram pairs in  $(s, t)$ . We can rewrite Eq. (2) as

$$\phi_r(s, t) = \sum_{\alpha_s \in \text{k-grams}(s)} \sum_{\alpha_t \in \text{k-grams}(t)} \bar{\phi}_r(\alpha_s, \alpha_t) \quad (3)$$

where  $\bar{\phi}_r(\alpha_s, \alpha_t) = \lambda^i$  if  $r$  (with  $i$  wildcards) matches  $(\alpha_s, \alpha_t)$ , otherwise  $\bar{\phi}_r(\alpha_s, \alpha_t) = 0$ .

For ease of computation, we re-write kb-SRK as

$$\begin{aligned} & \mathbb{K}_k((s_1, t_1), (s_2, t_2)) \\ = & \sum_{\substack{\alpha_{s_1} \in \text{k-grams}(s_1) \\ \alpha_{t_1} \in \text{k-grams}(t_1)}} \sum_{\substack{\alpha_{s_2} \in \text{k-grams}(s_2) \\ \alpha_{t_2} \in \text{k-grams}(t_2)}} \bar{\mathbb{K}}_k((\alpha_{s_1}, \alpha_{t_1}), (\alpha_{s_2}, \alpha_{t_2})) \end{aligned} \quad (4)$$

where

$$\bar{\mathbb{K}}_k = \sum_{r \in \mathbf{R}} \bar{\phi}_r(\alpha_{s_1}, \alpha_{t_1}) \bar{\phi}_r(\alpha_{s_2}, \alpha_{t_2}) \quad (5)$$

## 5.2 Algorithm for Computing Kernel

A straightforward computation of kb-SRK would be intractable. The computation of  $\mathbb{K}_k$  in Eq. (4) needs computations of  $\bar{\mathbb{K}}_k$  conducted  $O((n - k + 1)^4)$  times, where  $n$  denotes the maximum length of strings. Furthermore, the computation of  $\bar{\mathbb{K}}_k$  in Eq. (5) needs to perform matching of all the re-writing rules with the two  $k$ -gram pairs  $(\alpha_{s_1}, \alpha_{t_1})$ ,  $(\alpha_{s_2}, \alpha_{t_2})$ , which has time complexity  $O(k!)$ .

In this section, we will introduce an efficient algorithm, which can compute  $\bar{\mathbb{K}}_k$  and  $\mathbb{K}_k$  with the time complexities of  $O(k)$  and  $O(kn^2)$ , respectively. The latter is verified empirically.

### 5.2.1 Transformation of Problem

For ease of manipulation, our method transforms the computation of kernel on  $k$ -grams into the computation on a new data structure called lists of doubles. We first explain how to make the transformation.

Suppose that  $\alpha_1, \alpha_2 \in \Sigma^k$  are  $k$ -grams, we use  $\alpha_1[i]$  and  $\alpha_2[i]$  to represent the  $i$ -th characters of them. We call a pair of characters a double. Thus  $\Sigma \times \Sigma$  denotes the set of doubles and  $\alpha_s^D, \alpha_t^D \in (\Sigma \times$

$$\begin{aligned} \alpha_{s_1} &= \text{abbccbb} ; & \alpha_{s_2} &= \text{abccdd} ; \\ \alpha_{t_1} &= \text{cbcbbcb} ; & \alpha_{t_2} &= \text{cbccdc} ; \end{aligned}$$

Figure 2: Example of two  $k$ -gram pairs.

$$\begin{aligned} \alpha_s^D &= (\text{a, a}), (\text{b, b}), (\mathbf{b, c}), (\text{c, c}), (\text{c, c}), (\mathbf{b, d}), (\mathbf{b, d}) \\ \alpha_t^D &= (\text{c, c}), (\text{b, b}), (\text{c, c}), (\mathbf{b, c}), (\mathbf{b, d}), (\text{c, c}), (\mathbf{b, d}) \end{aligned}$$

Figure 3: Example of the pair of double lists combined from the two  $k$ -gram pairs in Fig. 2. Non-identical doubles are in bold.

$\Sigma)^k$  denote lists of doubles. The following operation combines two  $k$ -grams into a list of doubles.

$$\alpha_1 \otimes \alpha_2 = ((\alpha_1[1], \alpha_2[1]), \dots, (\alpha_1[k], \alpha_2[k])).$$

We denote  $\alpha_1 \otimes \alpha_2[i]$  as the  $i$ -th element of the list. Fig. 3 shows example lists of doubles combined from  $k$ -grams.

We introduce the set of *identical doubles*  $\mathbf{I} = \{(c, c) \mid c \in \Sigma\}$  and the set of *non-identical doubles*  $\mathbf{N} = \{(c, c') \mid c, c' \in \Sigma \text{ and } c \neq c'\}$ . Obviously,  $\mathbf{I} \cup \mathbf{N} = \Sigma \times \Sigma$  and  $\mathbf{I} \cap \mathbf{N} = \emptyset$ .

We define the set of *re-writing rules for double lists*  $\mathbf{R}^D = \{r^D = (\beta_s^D, \beta_t^D, \tau) \mid \beta_s^D, \beta_t^D \in (\mathbf{I} \cup \{?\})^k, \tau \text{ is a bijective alignment}\}$  where  $\beta_s^D$  and  $\beta_t^D$  are lists of identical doubles including wildcards and with length  $k$ . We say rule  $r^D$  matches a pair of double lists  $(\alpha_s^D, \alpha_t^D)$  iff.  $\beta_s^D, \beta_t^D$  can be changed into  $\alpha_s^D$  and  $\alpha_t^D$  by substituting each wildcard pair to a double in  $\Sigma \times \Sigma$ , and the double substituting the wildcard pair  $\beta_s^D[i]$  and  $\beta_t^D[j]$  must be an identical double when there is an alignment  $(i, j) \in \tau$ . The rule set defined here and the rule set in Sec. 4 only differ on the elements where re-writing occurs. Fig. 4 (B) shows an example of re-writing rule for double lists. The pair of double lists in Fig. 3 can match with the re-writing rule.

### 5.2.2 Computing $\bar{\mathbb{K}}_k$

We consider how to compute  $\bar{\mathbb{K}}_k$  by extending the computation from  $k$ -grams to double lists.

The following lemma shows that computing the weighted sum of re-writing rules matching  $k$ -gram pairs  $(\alpha_{s_1}, \alpha_{t_1})$  and  $(\alpha_{s_2}, \alpha_{t_2})$  is equivalent to computing the weighted sum of re-writing rules for double lists matching  $(\alpha_{s_1} \otimes \alpha_{s_2}, \alpha_{t_1} \otimes \alpha_{t_2})$ .



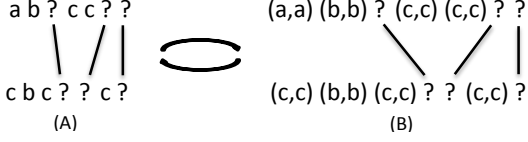


Figure 4: For re-writing rule (A) matching both  $k$ -gram pairs shown in Fig. 2, there is a corresponding re-writing rule for double lists (B) matching the pair of double lists shown in Fig. 3.

$$\begin{aligned} \#_{\Sigma \times \Sigma}(\alpha_s^D) &= \{(a, a): 1, (b, b): 1, (\mathbf{b}, \mathbf{c}): 1, (\mathbf{b}, \mathbf{d}): 2, (c, c): 2\} \\ \#_{\Sigma \times \Sigma}(\alpha_t^D) &= \{(a, a): 0, (b, b): 1, (\mathbf{b}, \mathbf{c}): 1, (\mathbf{b}, \mathbf{d}): 2, (c, c): 3\} \end{aligned}$$

Figure 5: Example of  $\#_{\Sigma \times \Sigma}(\cdot)$  for the two double lists shown in Fig. 3. Doubles not appearing in both  $\alpha_s^D$  and  $\alpha_t^D$  are not shown.

**Lemma 1.** *For any two  $k$ -gram pairs  $(\alpha_{s_1}, \alpha_{t_1})$  and  $(\alpha_{s_2}, \alpha_{t_2})$ , there exists a one-to-one mapping from the set of re-writing rules matching them to the set of re-writing rules matching the corresponding double lists  $(\alpha_{s_1} \otimes \alpha_{s_2}, \alpha_{t_1} \otimes \alpha_{t_2})$ .*

The re-writing rule in Fig. 4 (A) matches the  $k$ -gram pairs in Fig. 2. Equivalently, the re-writing rule for double lists in Fig. 4 (B) matches the pair of double lists in Fig. 3. By lemma 1 and Eq. 5, we have

$$\bar{K}_k = \sum_{r^D \in R^D} \bar{\phi}_{r^D}(\alpha_{s_1} \otimes \alpha_{s_2}, \alpha_{t_1} \otimes \alpha_{t_2}) \quad (6)$$

where  $\bar{\phi}_{r^D}(\alpha_s^D, \alpha_t^D) = \lambda^{2i}$  if the rewriting rule for double lists  $r^D$  with  $i$  wildcards matches  $(\alpha_s^D, \alpha_t^D)$ , otherwise  $\bar{\phi}_{r^D}(\alpha_s^D, \alpha_t^D) = 0$ . To get  $\bar{K}_k$ , we just need to compute the weighted sum of re-writing rules for double lists matching  $(\alpha_{s_1} \otimes \alpha_{s_2}, \alpha_{t_1} \otimes \alpha_{t_2})$ . Thus, we can work on the ‘‘combined’’ pair of double lists instead of two pairs of  $k$ -grams.

Instead of enumerating all possible re-writing rules and checking whether they can match the given pair of double lists, we only calculate the number of possibilities of ‘‘generating’’ from the pair of double lists to the re-writing rules matching it, which can be carried out efficiently. We say that a re-writing rule of double lists can be generated from a pair of double lists  $(\alpha_s^D, \alpha_t^D)$ , if they match with each other. From the definition of  $R^D$ , in each generation, the identical doubles in  $\alpha_s^D$  and  $\alpha_t^D$  can be either or not substituted by an aligned wildcard pair in the re-writing

---

### Algorithm 1: Computing $\bar{K}_k$

---

**Input:**  $k$ -gram pair  $(\alpha_{s_1}, \alpha_{t_1})$  and  $(\alpha_{s_2}, \alpha_{t_2})$

**Output:**  $\bar{K}_k((\alpha_{s_1}, \alpha_{t_1}), (\alpha_{s_2}, \alpha_{t_2}))$

- 1 Set  $(\alpha_s^D, \alpha_t^D) = (\alpha_{s_1} \otimes \alpha_{s_2}, \alpha_{t_1} \otimes \alpha_{t_2})$ ;
  - 2 Compute  $\#_{\Sigma \times \Sigma}(\alpha_s^D)$  and  $\#_{\Sigma \times \Sigma}(\alpha_t^D)$ ;
  - 3  $result = 1$ ;
  - 4 **for** each  $e \in \Sigma \times \Sigma$  satisfies  $\#_e(\alpha_s^D) + \#_e(\alpha_t^D) \neq 0$  **do**
  - 5      $g_e = 0, n_e = \min\{\#_e(\alpha_s^D), \#_e(\alpha_t^D)\}$ ;
  - 6     **for**  $0 \leq i \leq n_e$  **do**
  - 7          $g_e = g_e + a_i^{(e)} \lambda^{2i}$ ;
  - 8      $result = result * g$ ;
  - 9 **return**  $result$ ;
- 

rule, and all the non-identical doubles in  $\alpha_s^D$  and  $\alpha_t^D$  must be substituted by aligned wildcard pairs. From this observation and Eq. 6,  $\bar{K}_k$  only depends on the number of times each double occurs in the double lists.

Let  $e$  be a double. We denote  $\#_e(\alpha^D)$  as the number of times  $e$  occurs in the list of doubles  $\alpha^D$ . Also, for a set of doubles  $S \subseteq \Sigma \times \Sigma$ , we denote  $\#_S(\alpha^D)$  as a vector in which each element represents  $\#_e(\alpha^D)$  of each double  $e \in S$ . We can find a function  $g$  such that

$$\bar{K}_k = g(\#_{\Sigma \times \Sigma}(\alpha_{s_1} \otimes \alpha_{s_2}), \#_{\Sigma \times \Sigma}(\alpha_{t_1} \otimes \alpha_{t_2})) \quad (7)$$

Alg. 1 shows how to compute  $\bar{K}_k$ .  $\#_{\Sigma \times \Sigma}(\cdot)$  is computed from the two pairs of  $k$ -grams in line 1-2. The final score is made through the iterative calculation on the two lists (lines 4-8).

The key of Alg. 1 is the calculation of  $g_e$  based on  $a_i^{(e)}$  (line 7). Here we use  $a_i^{(e)}$  to denote the number of possibilities for which  $i$  pairs of aligned wildcards can be generated from  $e$  in both  $\alpha_s^D$  and  $\alpha_t^D$ .  $a_i^{(e)}$  can be computed as follows.

(1) If  $e \in N$  and  $\#_e(\alpha_s^D) \neq \#_e(\alpha_t^D)$ , then  $a_i^{(e)} = 0$  for any  $i$ .

(2) If  $e \in N$  and  $\#_e(\alpha_s^D) = \#_e(\alpha_t^D) = j$ , then  $a_j^{(e)} = j!$  and  $a_i^{(e)} = 0$  for any  $i \neq j$ .

(3) If  $e \in I$ , then  $a_i^{(e)} = \binom{\#_e(\alpha_s^D)}{i} \binom{\#_e(\alpha_t^D)}{i} i!$ .

We next explain the rationale behind the above computations. In (1), since  $\#_e(\alpha_s^D) \neq \#_e(\alpha_t^D)$ , it is impossible to generate a re-writing rule in which all

the occurrences of non-identical double  $e$  are substituted by pairs of aligned wildcards. In (2),  $j$  pairs of aligned wildcards can be generated from all the occurrences of non-identical double  $e$  in both  $\alpha_s^D$  and  $\alpha_t^D$ . The number of combinations thus is  $j!$ . In (3), a pair of aligned wildcards can either be generated or not from a pair of identical doubles in  $\alpha_s^D$  and  $\alpha_t^D$ . We can select  $i$  occurrences of identical double  $e$  from  $\alpha_s^D$ ,  $i$  occurrences from  $\alpha_t^D$ , and generate all possible aligned wildcards from them.

In the loop of lines 4-8, we only need to consider  $a_i^{(e)}$  for  $0 \leq i \leq \min\{\#_e(\alpha_s^D), \#_e(\alpha_t^D)\}$ , because  $a_i^{(e)} = 0$  for the rest of  $i$ .

To sum up, Eq. 7 can be computed as below, which is exactly the computation at lines 3-8.

$$g(\#_{\Sigma \times \Sigma}(\alpha_s^D), \#_{\Sigma \times \Sigma}(\alpha_t^D)) = \prod_{e \in \Sigma \times \Sigma} \left( \sum_{i=0}^{n_e} a_i^{(e)} \lambda^{2i} \right) \quad (8)$$

For the  $k$ -gram pairs in Fig. 2, we first create lists of doubles in Fig. 3 and compute  $\#_{\Sigma \times \Sigma}(\cdot)$  for them (lines 1-2 of Alg. 1), as shown in Fig. 5. We next compute  $K_k$  from  $\#_{\Sigma \times \Sigma}(\alpha_s^D)$  and  $\#_{\Sigma \times \Sigma}(\alpha_t^D)$  in Fig. 5 (lines 3-8 of Alg. 1) and obtain  $K_k = (1)(1 + \lambda^2)(\lambda^2)(2\lambda^4)(1 + 6\lambda^2 + 6\lambda^4) = 12\lambda^{12} + 24\lambda^{10} + 14\lambda^8 + 2\lambda^6$ .

### 5.2.3 Computing $K_k$

Algorithm 2 shows how to compute  $K_k$ . It prepares two maps  $m_s$  and  $m_t$  and two vectors of counters  $c_s$  and  $c_t$ . In  $m_s$  and  $m_t$ , each key  $\#_N(\cdot)$  maps a set of values  $\#_{\Sigma \times \Sigma}(\cdot)$ . Counters  $c_s$  and  $c_t$  count the frequency of each  $\#_{\Sigma \times \Sigma}(\cdot)$ . Recall that  $\#_N(\alpha_{s_1} \otimes \alpha_{s_2})$  denotes a vector whose element is  $\#_e(\alpha_{s_1} \otimes \alpha_{s_2})$  for  $e \in N$ .  $\#_{\Sigma \times \Sigma}(\alpha_{s_1} \otimes \alpha_{s_2})$  denotes a vector whose element is  $\#_e(\alpha_{s_1} \otimes \alpha_{s_2})$  where  $e$  is any possible double.

One can easily verify the output of the algorithm is exactly the value of  $K_k$ . First,  $\bar{K}_k((\alpha_{s_1}, \alpha_{t_1}), (\alpha_{s_2}, \alpha_{t_2})) = 0$  if  $\#_N(\alpha_{s_1} \otimes \alpha_{s_2}) \neq \#_N(\alpha_{t_1} \otimes \alpha_{t_2})$ . Therefore, we only need to consider those  $\alpha_{s_1} \otimes \alpha_{s_2}$  and  $\alpha_{t_1} \otimes \alpha_{t_2}$  which have the same key (lines 10-13). We group the  $k$ -gram pairs by their key in lines 2-5 and lines 6-9.

Moreover, the following relation holds

$$\bar{K}_k((\alpha_{s_1}, \alpha_{t_1}), (\alpha_{s_2}, \alpha_{t_2})) = \bar{K}_k((\alpha'_{s_1}, \alpha'_{t_1}), (\alpha'_{s_2}, \alpha'_{t_2}))$$

if  $\#_{\Sigma \times \Sigma}(\alpha_{s_1} \otimes \alpha_{s_2}) = \#_{\Sigma \times \Sigma}(\alpha'_{s_1} \otimes \alpha'_{s_2})$  and  $\#_{\Sigma \times \Sigma}(\alpha_{t_1} \otimes \alpha_{t_2}) = \#_{\Sigma \times \Sigma}(\alpha'_{t_1} \otimes \alpha'_{t_2})$ , where  $\alpha'_{s_1}, \alpha'_{s_2}, \alpha'_{t_1}, \alpha'_{t_2}$  are

---

### Algorithm 2: Computing $K_k$

---

**Input:** string pair  $(s_1, t_1)$  and  $(s_2, t_2)$ , window size  $k$

**Output:**  $K_k((s_1, t_1), (s_2, t_2))$

```

1 Initialize two maps  $m_s$  and  $m_t$  and two counters  $c_s$  and  $c_t$ ;
2 for each  $k$ -gram  $\alpha_{s_1}$  in  $s_1$  do
3   for each  $k$ -gram  $\alpha_{s_2}$  in  $s_2$  do
4     Update  $m_s$  with key-value pair  $(\#_N(\alpha_{s_1} \otimes \alpha_{s_2}), \#_{\Sigma \times \Sigma}(\alpha_{s_1} \otimes \alpha_{s_2}))$ ;
5      $c_s[\#_{\Sigma \times \Sigma}(\alpha_{s_1} \otimes \alpha_{s_2})]++$ ;
6 for each  $k$ -gram  $\alpha_{t_1}$  in  $t_1$  do
7   for each  $k$ -gram  $\alpha_{t_2}$  in  $t_2$  do
8     Update  $m_t$  with key-value pair  $(\#_N(\alpha_{t_1} \otimes \alpha_{t_2}), \#_{\Sigma \times \Sigma}(\alpha_{t_1} \otimes \alpha_{t_2}))$ ;
9      $c_t[\#_{\Sigma \times \Sigma}(\alpha_{t_1} \otimes \alpha_{t_2})]++$ ;
10 for each key  $\in m_s.keys \cap m_t.keys$  do
11   for each  $v_s \in m_s[key]$  do
12     for each  $v_t \in m_t[key]$  do
13       result +=  $c_s[v_s]c_t[v_t]g(v_s, v_t)$ ;
14 return result;
```

---

other  $k$ -grams. Therefore, we only need to take  $\#_{\Sigma \times \Sigma}(\alpha_{s_1} \otimes \alpha_{s_2})$  and  $\#_{\Sigma \times \Sigma}(\alpha_{t_1} \otimes \alpha_{t_2})$  as the value under each key and count its frequency. That is to say,  $\#_{\Sigma \times \Sigma}$  provides sufficient statistics for computing  $\bar{K}_k$ .

The quantity  $g(v_s, v_t)$  in line 13 is computed by Alg. 1 (lines 3-8).

### 5.3 Time Complexity

The time complexities of Alg. 1 and Alg. 2 are shown below.

For Alg. 1, lines 1-2 can be executed in  $O(k)$ . The time for executing line 7 is less than  $\#_e(\alpha_s^D) + \#_e(\alpha_t^D) + 1$  for each  $e$  satisfying  $\#_e(\alpha_s^D) \neq 0$  or  $\#_e(\alpha_t^D) \neq 0$ . Since  $\sum_{e \in \Sigma \times \Sigma} \#_e(\alpha_s^D) = \sum_{e \in \Sigma \times \Sigma} \#_e(\alpha_t^D) = k$ , the time for executing lines 3-8 is less than  $4k$ , which results in the  $O(k)$  time complexity of Alg. 1.

For Alg. 2, we denote  $n = \max\{|s_1|, |s_2|, |t_1|, |t_2|\}$ . It is easy to see that if the maps and counters in the algorithm are implemented by hash maps, the time complexities of lines 2-5 and lines 6-9 are  $O(kn^2)$ . However, analyzing the time complexity of lines 10-

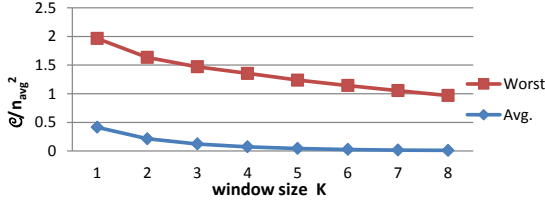


Figure 6: Relation between ratio  $\mathcal{C}/n_{avg}^2$  and window size  $k$  when running Alg. 2 on MSR Paraphrases Corpus.

13 is quite difficult.

Lemma 2 and Theorem 1 provide an upper bound of the number of times computing  $g(v_s, v_t)$  in line 13, denoted as  $\mathcal{C}$ .

**Lemma 2.** For  $\alpha_{s_1} \in k\text{-grams}(s_1)$  and  $\alpha_{s_2}, \alpha'_{s_2} \in k\text{-grams}(s_2)$ , we have  $\#\Sigma \times \Sigma(\alpha_{s_1} \otimes \alpha_{s_2}) = \#\Sigma \times \Sigma(\alpha_{s_1} \otimes \alpha'_{s_2})$  if  $\#_N(\alpha_{s_1} \otimes \alpha_{s_2}) = \#_N(\alpha_{s_1} \otimes \alpha'_{s_2})$ .

**Theorem 1.**  $\mathcal{C}$  is  $O(n^3)$ .

By Lemma 2, each  $m_s[key]$  contains at most  $n - k + 1$  elements. Together with the fact that  $\sum_{key} m_s[key] = (n - k + 1)^2$ , Theorem 1 is proved. It can be also proved that  $\mathcal{C}$  is  $O(n^2)$  when  $k = 1$ .

Empirical study shows that  $O(n^3)$  is a loose upper bound for  $\mathcal{C}$ . Let  $n_{avg}$  denote the average length of  $s_1, t_1, s_2$  and  $t_2$ . Our experiment on all pairs of sentences on MSR Paraphrase (Fig. 6) shows that  $\mathcal{C}$  is in the same order of  $n_{avg}^2$  in the worst case and  $\mathcal{C}/n_{avg}^2$  decreases with increasing  $k$  in both average case and worst case, which indicates that  $\mathcal{C}$  is  $O(n^2)$  and the overall time complexity of Alg. 2 is  $O(kn^2)$ .

## 6 Experiments

We evaluated the performances of the three types of string re-writing kernels on paraphrase identification and recognizing textual entailment: pairwise  $k$ -spectrum kernel (ps-SRK), pairwise  $k$ -wildcard kernel (pw-SRK), and  $k$ -gram bijective string re-writing kernel (kb-SRK). We set  $\lambda = 1$  for all kernels. The performances were measured by accuracy (e.g. percentage of correct classifications).

In both experiments, we used LIBSVM with default parameters (Chang et al., 2011) as the classifier. All the sentences in the training and test sets were segmented into words by the tokenizer at OpenNLP (Baldrige et al., ). We further conducted stemming on the words with IVEONIK English Stemmer (<http://www.iveonik.com/>).

We normalized each kernel by  $\tilde{K}(x, y) = \frac{K(x, y)}{\sqrt{K(x, x)K(y, y)}}$  and then tried them under different window sizes  $k$ . We also tried to combine the kernels with two lexical features “unigram precision and recall” proposed in (Wan et al., 2006), referred to as PR. For each kernel  $K$ , we tested the window size settings of  $K_1 + \dots + K_{k_{max}}$  ( $k_{max} \in \{1, 2, 3, 4\}$ ) together with the combination with PR and we report the best accuracies of them in Tab 1 and Tab 2.

### 6.1 Paraphrase Identification

The task of paraphrase identification is to examine whether two sentences have the same meaning. We trained and tested all the methods on the MSR Paraphrase Corpus (Dolan and Brockett, 2005; Quirk et al., 2004) consisting of 4,076 sentence pairs for training and 1,725 sentence pairs for testing.

The experimental results on different SRKs are shown in Table 1. It can be seen that kb-SRK outperforms ps-SRK and pw-SRK. The results by the state-of-the-art methods reported in previous work are also included in Table 1. kb-SRK outperforms the existing lexical approach (Zhang and Patrick, 2005) and kernel approach (Lintean and Rus, 2011). It also works better than the other approaches listed in the table, which use syntactic trees or dependency relations.

Fig. 7 gives detailed results of the kernels under different maximum  $k$ -gram lengths  $k_{max}$  with and without PR. The results of ps-SRK and pw-SRK without combining PR under different  $k$  are all below 71%, therefore they are not shown for clar-

Method	Acc.
Zhang and Patrick (2005)	71.9
Lintean and Rus (2011)	73.6
Heilman and Smith (2010)	73.2
Qiu et al. (2006)	72.0
Wan et al. (2006)	75.6
Das and Smith (2009)	73.9
Das and Smith (2009)(PoE)	76.1
Our baseline (PR)	73.6
Our method (ps-SRK)	75.6
Our method (pw-SRK)	75.0
Our method (kb-SRK)	<b>76.3</b>

Table 1: Comparison with state-of-the-arts on MSRP.

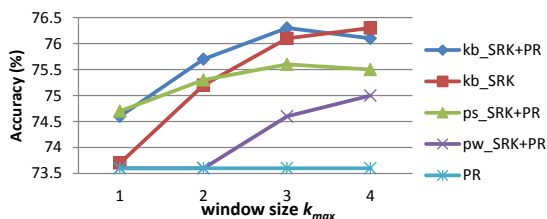


Figure 7: Performances of different kernels under different maximum window size  $k_{max}$  on MSRP.

ity. By comparing the results of kb-SRK and pw-SRK we can see that the bijective property in kb-SRK is really helpful for improving the performance (note that both methods use wildcards). Furthermore, the performances of kb-SRK with and without combining PR increase dramatically with increasing  $k_{max}$  and reach the peaks (better than state-of-the-art) when  $k_{max}$  is four, which shows the power of the lexical and structural similarity captured by kb-SRK.

## 6.2 Recognizing Textual Entailment

Recognizing textual entailment is to determine whether a sentence (sometimes a short paragraph) can entail the other sentence (Giampiccolo et al., 2007). RTE-3 is a widely used benchmark dataset. Following the common practice, we combined the development set of RTE-3 and the whole datasets of RTE-1 and RTE-2 as training data and took the test set of RTE-3 as test data. The train and test sets contain 3,767 and 800 sentence pairs.

The results are shown in Table 2. Again, kb-SRK outperforms ps-SRK and pw-SRK. As indicated in (Heilman and Smith, 2010), the top-performing RTE systems are often built with significant engi-

Method	Acc.
Harmeling (2007)	59.5
de Marneffe et al. (2006)	60.5
M&M, (2007) (NL)	59.4
M&M, (2007) (Hybrid)	64.3
Zanzotto et al. (2007)	<b>65.75</b>
Heilman and Smith (2010)	62.8
Our baseline (PR)	62.0
Our method (ps-SRK)	64.6
Our method (pw-SRK)	63.8
Our method (kb-SRK)	65.1

Table 2: Comparison with state-of-the-arts on RTE-3.

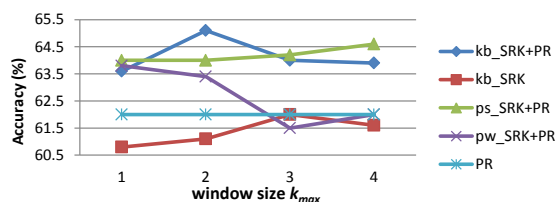


Figure 8: Performances of different kernels under different maximum window size  $k_{max}$  on RTE-3.

neering efforts. Therefore, we only compare with the six systems which involves less engineering. kb-SRK still outperforms most of those state-of-the-art methods even if it does not exploit any other lexical semantic sources and syntactic analysis tools.

Fig. 8 shows the results of the kernels under different parameter settings. Again, the results of ps-SRK and pw-SRK without combining PR are too low to be shown (all below 55%). We can see that PR is an effective method for this dataset and the overall performances are substantially improved after combining it with the kernels. The performance of kb-SRK reaches the peak when window size becomes two.

## 7 Conclusion

In this paper, we have proposed a novel class of kernel functions for sentence re-writing, called string re-writing kernel (SRK). SRK measures the lexical and structural similarity between two pairs of sentences without using syntactic trees. The approach is theoretically sound and is flexible to formulations of sentences. A specific instance of SRK, referred to as kb-SRK, has been developed which can balance the effectiveness and efficiency for sentence re-writing. Experimental results show that kb-SRK achieve better results than state-of-the-art methods on paraphrase identification and recognizing textual entailment.

## Acknowledgments

This work is supported by the National Basic Research Program (973 Program) No. 2012CB316301.

## References

Baldrige, J. , Morton, T. and Bierner G. *OpenNLP*. <http://opennlp.sourceforge.net/>.

- Barzilay, R. and Lee, L. 2003. *Learning to paraphrase: An unsupervised approach using multiple-sequence alignment*. Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, pp. 16–23.
- Basilico, J. and Hofmann, T. 2004. *Unifying collaborative and content-based filtering*. Proceedings of the twenty-first international conference on Machine learning, pp. 9, 2004.
- Ben-Hur, A. and Noble, W.S. 2005. *Kernel methods for predicting protein–protein interactions*. Bioinformatics, vol. 21, pp. i38–i46, Oxford Univ Press.
- Bhagat, R. and Ravichandran, D. 2008. *Large scale acquisition of paraphrases for learning surface patterns*. Proceedings of ACL-08: HLT, pp. 674–682.
- Chang, C. and Lin, C. 2011. *LIBSVM: A library for support vector machines*. ACM Transactions on Intelligent Systems and Technology vol. 2, issue 3, pp. 27:1–27:27. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
- Das, D. and Smith, N.A. 2009. *Paraphrase identification as probabilistic quasi-synchronous recognition*. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp. 468–476.
- de Marneffe, M., MacCartney, B., Grenager, T., Cer, D., Rafferty A. and Manning C.D. 2006. *Learning to distinguish valid textual entailments*. Proc. of the Second PASCAL Challenges Workshop.
- Dolan, W.B. and Brockett, C. 2005. *Automatically constructing a corpus of sentential paraphrases*. Proc. of IWP.
- Giampiccolo, D., Magnini B., Dagan I., and Dolan B., editors 2007. *The third pascal recognizing textual entailment challenge*. Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, pp. 1–9.
- Harmeling, S. 2007. *An extensible probabilistic transformation-based approach to the third recognizing textual entailment challenge*. Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, pp. 137–142, 2007.
- Heilman, M. and Smith, N.A. 2010. *Tree edit models for recognizing textual entailments, paraphrases, and answers to questions*. Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 1011-1019.
- Kashima, H. , Oyama, S. , Yamanishi, Y. and Tsuda, K. 2009. *On pairwise kernels: An efficient alternative and generalization analysis*. Advances in Knowledge Discovery and Data Mining, pp. 1030-1037, 2009, Springer.
- Kimeldorf, G. and Wahba, G. 1971. *Some results on Tchebycheffian spline functions*. Journal of Mathematical Analysis and Applications, Vol.33, No.1, pp.82-95, Elsevier.
- Lin, D. and Pantel, P. 2001. *DIRT-discovery of inference rules from text*. Proc. of ACM SIGKDD Conference on Knowledge Discovery and Data Mining.
- Lintean, M. and Rus, V. 2011. *Dissimilarity Kernels for Paraphrase Identification*. Twenty-Fourth International FLAIRS Conference.
- Leslie, C. , Eskin, E. and Noble, W.S. 2002. *The spectrum kernel: a string kernel for SVM protein classification*. Pacific symposium on biocomputing vol. 575, pp. 564-575, Hawaii, USA.
- Leslie, C. and Kuang, R. 2004. *Fast string kernels using inexact matching for protein sequences*. The Journal of Machine Learning Research vol. 5, pp. 1435-1455.
- Lodhi, H. , Saunders, C. , Shawe-Taylor, J. , Cristianini, N. and Watkins, C. 2002. *Text classification using string kernels*. The Journal of Machine Learning Research vol. 2, pp. 419-444.
- MacCartney, B. and Manning, C.D. 2008. *Modeling semantic containment and exclusion in natural language inference*. Proceedings of the 22nd International Conference on Computational Linguistics, vol. 1, pp. 521-528, 2008.
- Moschitti, A. and Zanzotto, F.M. 2007. *Fast and Effective Kernels for Relational Learning from Texts*. Proceedings of the 24th Annual International Conference on Machine Learning, Corvallis, OR, USA, 2007.
- Qiu, L. and Kan, M.Y. and Chua, T.S. 2006. *Paraphrase recognition via dissimilarity significance classification*. Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, pp. 18–26.
- Quirk, C. , Brockett, C. and Dolan, W. 2004. *Monolingual machine translation for paraphrase generation*. Proceedings of EMNLP 2004, pp. 142-149, Barcelona, Spain.
- Schölkopf, B. and Smola, A.J. 2002. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. The MIT Press, Cambridge, MA.
- Vapnik, V.N. 2000. *The nature of statistical learning theory*. Springer Verlag.
- Wan, S. , Dras, M. , Dale, R. and Paris, C. 2006. *Using dependency-based features to take the “Para-farce” out of paraphrase*. Proc. of the Australasian Language Technology Workshop, pp. 131–138.
- Zanzotto, F.M. , Pennacchiotti, M. and Moschitti, A. 2007. *Shallow semantics in fast textual entailment*

*rule learners*. Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing, pp. 72–77.

Zhang, Y. and Patrick, J. 2005. *Paraphrase identification by text canonicalization*. Proceedings of the Australasian Language Technology Workshop, pp. 160–166.

# Translation Model Adaptation for Statistical Machine Translation with Monolingual Topic Information\*

Jinsong Su<sup>1,2</sup>, Hua Wu<sup>3</sup>, Haifeng Wang<sup>3</sup>, Yidong Chen<sup>1</sup>, Xiaodong Shi<sup>1</sup>,  
Huailin Dong<sup>1</sup>, and Qun Liu<sup>2</sup>

Xiamen University, Xiamen, China<sup>1</sup>

Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China<sup>2</sup>

Baidu Inc., Beijing, China<sup>3</sup>

{jssu, ydchen, mandel, hldong}@xmu.edu.cn

{wu.hua, wanghaifeng}@baicu.com

liuqun@ict.ac.cn

## Abstract

To adapt a translation model trained from the data in one domain to another, previous works paid more attention to the studies of parallel corpus while ignoring the in-domain monolingual corpora which can be obtained more easily. In this paper, we propose a novel approach for translation model adaptation by utilizing in-domain monolingual topic information instead of the in-domain bilingual corpora, which incorporates the topic information into translation probability estimation. Our method establishes the relationship between the out-of-domain bilingual corpus and the in-domain monolingual corpora via a topic mapping and phrase-topic distribution probability estimation from in-domain monolingual corpora. Experimental result on the NIST Chinese-English translation task shows that our approach significantly outperforms the baseline system.

## 1 Introduction

In recent years, statistical machine translation(SMT) has been rapidly developing with more and more novel translation models being proposed and put into practice (Koehn et al., 2003; Och and Ney, 2004; Galley et al., 2006; Liu et al., 2006; Chiang, 2007; Chiang, 2010). However, similar to other natural language processing(NLP) tasks, SMT systems often suffer from domain adaptation problem during practical applications. The simple reason is that the underlying statistical models always tend to closely

approximate the empirical distributions of the training data, which typically consist of bilingual sentences and monolingual target language sentences. When the translated texts and the training data come from the same domain, SMT systems can achieve good performance, otherwise the translation quality degrades dramatically. Therefore, it is of significant importance to develop translation systems which can be effectively transferred from one domain to another, for example, from *newswire* to *weblog*.

According to adaptation emphases, domain adaptation in SMT can be classified into translation model adaptation and language model adaptation. Here we focus on how to adapt a translation model, which is trained from the large-scale out-of-domain bilingual corpus, for domain-specific translation task, leaving others for future work. In this aspect, previous methods can be divided into two categories: one paid attention to collecting more sentence pairs by information retrieval technology (Hildebrand et al., 2005) or synthesized parallel sentences (Ueffing et al., 2008; Wu et al., 2008; Bertoldi and Federico, 2009; Schwenk and Senellart, 2009), and the other exploited the full potential of existing parallel corpus in a *mixture-modeling* (Foster and Kuhn, 2007; Civera and Juan, 2007; Lv et al., 2007) framework. However, these approaches focused on the studies of bilingual corpus synthesis and exploitation while ignoring the monolingual corpora, therefore limiting the potential of further translation quality improvement.

In this paper, we propose a novel adaptation method to adapt the translation model for domain-specific translation task by utilizing in-domain

\*Part of this work was done during the first author's internship at Baidu.

monolingual corpora. Our approach is inspired by the recent studies (Zhao and Xing, 2006; Zhao and Xing, 2007; Tam et al., 2007; Gong and Zhou, 2010; Ruiz and Federico, 2011) which have shown that a particular translation always appears in some specific topical contexts, and the topical context information has a great effect on translation selection. For example, “*bank*” often occurs in the sentences related to the *economy* topic when translated into “*yínháng*”, and occurs in the sentences related to the *geography* topic when translated to “*hén*”. Therefore, the co-occurrence frequency of the phrases in some specific context can be used to constrain the translation candidates of phrases. In a monolingual corpus, if “*bank*” occurs more often in the sentences related to the *economy* topic than the ones related to the *geography* topic, it is more likely that “*bank*” is translated to “*yínháng*” than to “*hén*”. With the out-of-domain bilingual corpus, we first incorporate the topic information into translation probability estimation, aiming to quantify the effect of the topical context information on translation selection. Then, we rescore all phrase pairs according to the phrase-topic and the word-topic posterior distributions of the additional in-domain monolingual corpora. As compared to the previous works, our method takes advantage of both the in-domain monolingual corpora and the out-of-domain bilingual corpus to incorporate the topic information into our translation model, thus breaking down the corpus barrier for translation quality improvement. The experimental results on the NIST data set demonstrate the effectiveness of our method.

The reminder of this paper is organized as follows: Section 2 provides a brief description of translation probability estimation. Section 3 introduces the adaptation method which incorporates the topic information into the translation model; Section 4 describes and discusses the experimental results; Section 5 briefly summarizes the recent related work about translation model adaptation. Finally, we end with a conclusion and the future work in Section 6.

## 2 Background

The statistical translation model, which contains phrase pairs with bi-directional *phrase probabilities* and bi-directional *lexical probabilities*, has a great

effect on the performance of SMT system. Phrase probability measures the co-occurrence frequency of a phrase pair, and lexical probability is used to validate the quality of the phrase pair by checking how well its words are translated to each other.

According to the definition proposed by (Koehn et al., 2003), given a source sentence  $\mathbf{f} = f_1^J = f_1, \dots, f_j, \dots, f_J$ , a target sentence  $\mathbf{e} = e_1^I = e_1, \dots, e_i, \dots, e_I$ , and its word alignment  $\mathbf{a}$  which is a subset of the Cartesian product of word positions:  $\mathbf{a} \subseteq (j, i) : j = 1, \dots, J; i = 1, \dots, I$ , the phrase pair  $(\tilde{f}, \tilde{e})$  is said to be *consistent* (Och and Ney, 2004) with the alignment if and only if: (1) there must be at least one word inside one phrase aligned to a word inside the other phrase and (2) no words inside one phrase can be aligned to a word outside the other phrase. After all consistent phrase pairs are extracted from training corpus, the phrase probabilities are estimated as *relative frequencies* (Och and Ney, 2004):

$$\phi(\tilde{e}|\tilde{f}) = \frac{\text{count}(\tilde{f}, \tilde{e})}{\sum_{\tilde{e}'} \text{count}(\tilde{f}, \tilde{e}')} \quad (1)$$

Here  $\text{count}(\tilde{f}, \tilde{e})$  indicates how often the phrase pair  $(\tilde{f}, \tilde{e})$  occurs in the training corpus.

To obtain the corresponding lexical weight, we first estimate a lexical translation probability distribution  $w(e|f)$  by *relative frequency* from the training corpus:

$$w(e|f) = \frac{\text{count}(f, e)}{\sum_{e'} \text{count}(f, e')} \quad (2)$$

Retaining the alignment  $\tilde{a}$  between the phrase pair  $(\tilde{f}, \tilde{e})$ , the corresponding lexical weight is calculated as

$$p_w(\tilde{e}|\tilde{f}, \tilde{a}) = \prod_{i=1}^{|\tilde{e}|} \frac{1}{|\{j|(j, i) \in \tilde{a}\}|} \sum_{\forall (j, i) \in \tilde{a}} w(e_i|f_j) \quad (3)$$

However, the above-mentioned method only counts the co-occurrence frequency of bilingual phrases, assuming that the translation probability is independent of the context information. Thus, the statistical model estimated from the training data is not suitable for text translation in different domains, resulting in a significant drop in translation quality.



### 3 Translation Model Adaptation via Monolingual Topic Information

In this section, we first briefly review the principle of Hidden Topic Markov Model (HTMM) which is the basis of our method, then describe our approach to translation model adaptation in detail.

#### 3.1 Hidden Topic Markov Model

During the last couple of years, topic models such as Probabilistic Latent Semantic Analysis (Hofmann, 1999) and Latent Dirichlet Allocation model (Blei, 2003), have drawn more and more attention and been applied successfully in NLP community. Based on the “bag-of-words” assumption that the order of words can be ignored, these methods model the text corpus by using a co-occurrence matrix of words and documents, and build generative models to infer the latent aspects or topics. Using these models, the words can be clustered into the derived topics with a probability distribution, and the correlation between words can be automatically captured via topics.

However, the “bag-of-words” assumption is an unrealistic oversimplification because it ignores the order of words. To remedy this problem, Gruber et al. (2007) propose HTMM, which models the topics of words in the document as a Markov chain. Based on the assumption that all words in the same sentence have the same topic and the successive sentences are more likely to have the same topic, HTMM incorporates the local dependency between words by Hidden Markov Model for better topic estimation.

HTMM can also be viewed as a soft clustering tool for words in training corpus. That is, HTMM can estimate the probability distribution of a topic over words, i.e. the topic-word distribution  $P(\text{word}|\text{topic})$  during training. Besides, HTMM derives inherent topics in sentences rather than in documents, so we can easily obtain the sentence-topic distribution  $P(\text{topic}|\text{sentence})$  in training corpus. Adopting maximum likelihood estimation (MLE), this posterior distribution makes it possible to effectively calculate the word-topic distribution  $P(\text{topic}|\text{word})$  and the phrase-topic distribution  $P(\text{topic}|\text{phrase})$  both of which are very important in our method.

#### 3.2 Adapted Phrase Probability Estimation

We utilize the additional in-domain monolingual corpora to adapt the out-of-domain translation model for domain-specific translation task. In detail, we build an adapted translation model in the following steps:

- Build a topic-specific translation model to quantify the effect of the topic information on the translation probability estimation.
- Estimate the topic posterior distributions of phrases in the in-domain monolingual corpora.
- Score the phrase pairs according to the predefined topic-specific translation model and the topic posterior distribution of phrases.

Formally, we incorporate monolingual topic information into translation probability estimation, and decompose the phrase probability  $\phi(\tilde{e}|\tilde{f})$ <sup>1</sup> as follows:

$$\begin{aligned}\phi(\tilde{e}|\tilde{f}) &= \sum_{t_f} \phi(\tilde{e}, t_f|\tilde{f}) \\ &= \sum_{t_f} \phi(\tilde{e}|\tilde{f}, t_f) \cdot P(t_f|\tilde{f})\end{aligned}\quad (4)$$

where  $\phi(\tilde{e}|\tilde{f}, t_f)$  indicates the probability of translating  $\tilde{f}$  into  $\tilde{e}$  given the source-side topic  $t_f$ ,  $P(t_f|\tilde{f})$  denotes the phrase-topic distribution of  $\tilde{f}$ .

To compute  $\phi(\tilde{e}|\tilde{f})$ , we first apply HTMM to respectively train two monolingual topic models with the following corpora: one is the source part of the out-of-domain bilingual corpus  $C_{f.out}$ , the other is the in-domain monolingual corpus  $C_{f.in}$  in the source language. Then, we respectively estimate  $\phi(\tilde{e}|\tilde{f}, t_f)$  and  $P(t_f|\tilde{f})$  from these two corpora. To avoid confusion, we further refine  $\phi(\tilde{e}|\tilde{f}, t_f)$  and  $P(t_f|\tilde{f})$  with  $\phi(\tilde{e}|\tilde{f}, t_{f.out})$  and  $P(t_{f.in}|\tilde{f})$ , respectively. Here,  $t_{f.out}$  is the topic clustered from the corpus  $C_{f.out}$ , and  $t_{f.in}$  represents the topic derived from the corpus  $C_{f.in}$ .

However, the two above-mentioned probabilities can not be directly multiplied in formula (4) because they are related to different topic spaces from

<sup>1</sup>Due to the limit of space, we omit the description of the calculation method of the phrase probability  $\phi(\tilde{f}|\tilde{e})$ , which can be adjusted in a similar way to  $\phi(\tilde{e}|\tilde{f})$  with the help of in-domain monolingual corpus in the target language.

different corpora. Besides, their topic dimensions are not assured to be the same. To solve this problem, we introduce the topic mapping probability  $P(t_{f\_out}|t_{f\_in})$  to map the in-domain phrase-topic distribution into the one in the out-domain topic space. To be specific, we obtain the out-of-domain phrase-topic distribution  $P(t_{f\_out}|\tilde{f})$  as follows:

$$P(t_{f\_out}|\tilde{f}) = \sum_{t_{f\_in}} P(t_{f\_out}|t_{f\_in}) \cdot P(t_{f\_in}|\tilde{f}) \quad (5)$$

Thus formula (4) can be further refined as the following formula:

$$\begin{aligned} \phi(\tilde{e}|\tilde{f}) &= \sum_{t_{f\_out}} \sum_{t_{f\_in}} \phi(\tilde{e}|\tilde{f}, t_{f\_out}) \\ &\quad \cdot P(t_{f\_out}|t_{f\_in}) \cdot P(t_{f\_in}|\tilde{f}) \quad (6) \end{aligned}$$

Next we will give detailed descriptions of the calculation methods for the three probability distributions mentioned in formula (6).

### 3.2.1 Topic-Specific Phrase Translation

**Probability**  $\phi(\tilde{e}|\tilde{f}, t_{f\_out})$

We follow the common practice (Koehn et al., 2003) to calculate the topic-specific phrase translation probability, and the only difference is that our method takes the topical context information into account when collecting the fractional counts of phrase pairs. With the sentence-topic distribution  $P(t_{f\_out}|\mathbf{f})$  from the relevant topic model of  $C_{f\_out}$ , the conditional probability  $\phi(\tilde{e}|\tilde{f}, t_{f\_out})$  can be easily obtained by MLE method:

$$\begin{aligned} &\phi(\tilde{e}|\tilde{f}, t_{f\_out}) \\ &= \frac{\sum_{\langle \mathbf{f}, \mathbf{e} \rangle \in C_{out}} count_{\langle \mathbf{f}, \mathbf{e} \rangle}(\tilde{f}, \tilde{e}) \cdot P(t_{f\_out}|\mathbf{f})}{\sum_{\tilde{e}'} \sum_{\langle \mathbf{f}, \mathbf{e} \rangle \in C_{out}} count_{\langle \mathbf{f}, \mathbf{e} \rangle}(\tilde{f}, \tilde{e}') \cdot P(t_{f\_out}|\mathbf{f})} \quad (7) \end{aligned}$$

where  $C_{out}$  is the out-of-domain bilingual training corpus, and  $count_{\langle \mathbf{f}, \mathbf{e} \rangle}(\tilde{f}, \tilde{e})$  denotes the number of the phrase pair  $(\tilde{f}, \tilde{e})$  in sentence pair  $\langle \mathbf{f}, \mathbf{e} \rangle$ .

### 3.2.2 Topic Mapping Probability $P(t_{f\_out}|t_{f\_in})$

Based on the two monolingual topic models respectively trained from  $C_{f\_in}$  and  $C_{f\_out}$ , we compute the topic mapping probability by using source word  $f$  as the pivot variable. Noticing that there

are some words occurring in one corpus only, we use the words belonging to both corpora during the mapping procedure. Specifically, we decompose  $P(t_{f\_out}|t_{f\_in})$  as follows:

$$\begin{aligned} &P(t_{f\_out}|t_{f\_in}) \\ &= \sum_{f \in C_{f\_out} \cap C_{f\_in}} P(t_{f\_out}|f) \cdot P(f|t_{f\_in}) \quad (8) \end{aligned}$$

Here we first get  $P(f|t_{f\_in})$  directly from the topic model related to  $C_{f\_in}$ . Then, considering the sentence-topic distribution  $P(t_{f\_out}|\mathbf{f})$  from the relevant topic model of  $C_{f\_out}$ , we define the word-topic distribution  $P(t_{f\_out}|f)$  as:

$$\begin{aligned} &P(t_{f\_out}|f) \\ &= \frac{P(t_{f\_out}|f) \sum_{\mathbf{f} \in C_{f\_out}} count_{\mathbf{f}}(f) \cdot P(t_{f\_out}|\mathbf{f})}{\sum_{t_{f\_out}} \sum_{\mathbf{f} \in C_{f\_out}} count_{\mathbf{f}}(f) \cdot P(t_{f\_out}|\mathbf{f})} \quad (9) \end{aligned}$$

where  $count_{\mathbf{f}}(f)$  denotes the number of the word  $f$  in sentence  $\mathbf{f}$ .

### 3.2.3 Phrase-Topic Distribution $P(t_{f\_in}|\tilde{f})$

A simple way to compute the phrase-topic distribution is to take the fractional counts from  $C_{f\_in}$  and then adopt MLE to obtain relative probability. However, it is infeasible in our model because some phrases occur in  $C_{f\_out}$  while being absent in  $C_{f\_in}$ . To solve this problem, we further compute this posterior distribution by the interpolation of two models:

$$\begin{aligned} P(t_{f\_in}|\tilde{f}) &= \theta \cdot P_{mle}(t_{f\_in}|\tilde{f}) + \\ &\quad (1 - \theta) \cdot P_{word}(t_{f\_in}|\tilde{f}) \quad (10) \end{aligned}$$

where  $P_{mle}(t_{f\_in}|\tilde{f})$  indicates the phrase-topic distribution by MLE,  $P_{word}(t_{f\_in}|\tilde{f})$  denotes the phrase-topic distribution which is decomposed into the topic posterior distribution at the word level, and  $\theta$  is the interpolation weight that can be optimized over the development data.

Given the number of the phrase  $\tilde{f}$  in sentence  $\mathbf{f}$  denoted as  $count_{\mathbf{f}}(\tilde{f})$ , we compute the in-domain phrase-topic distribution in the following way:

$$\begin{aligned} &P_{mle}(t_{f\_in}|\tilde{f}) \\ &= \frac{\sum_{\mathbf{f} \in C_{f\_in}} count_{\mathbf{f}}(\tilde{f}) \cdot P(t_{f\_in}|\mathbf{f})}{\sum_{t_{f\_in}} \sum_{\mathbf{f} \in C_{f\_in}} count_{\mathbf{f}}(\tilde{f}) \cdot P(t_{f\_in}|\mathbf{f})} \quad (11) \end{aligned}$$

Under the assumption that the topics of all words in the same phrase are independent, we consider two methods to calculate  $P_{word}(t_{f.in}|\tilde{f})$ . One is a “Noisy-OR” combination method (Zens and Ney, 2004) which has shown good performance in calculating similarities between bags-of-words in different languages. Using this method,  $P_{word}(t_{f.in}|\tilde{f})$  is defined as:

$$\begin{aligned} & P_{word}(t_{f.in}|\tilde{f}) \\ = & 1 - P_{word}(\bar{t}_{f.in}|\tilde{f}) \\ \approx & 1 - \prod_{f_j \in \tilde{f}} P(\bar{t}_{f.in}|f_j) \\ = & 1 - \prod_{f_j \in \tilde{f}} (1 - P(t_{f.in}|f_j)) \end{aligned} \quad (12)$$

where  $P_{word}(\bar{t}_{f.in}|\tilde{f})$  represents the probability that  $t_{f.in}$  is not the topic of the phrase  $\tilde{f}$ . Similarly,  $P(\bar{t}_{f.in}|f_j)$  indicates the probability that  $t_{f.in}$  is not the topic of the word  $f_j$ .

The other method is an “Averaging” combination one. With the assumption that  $t_{f.in}$  is the topic of  $\tilde{f}$  if at least one of the words in  $\tilde{f}$  belongs to this topic, we derive  $P_{word}(t_{f.in}|\tilde{f})$  as follows:

$$P_{word}(t_{f.in}|\tilde{f}) \approx \sum_{f_j \in \tilde{f}} P(t_{f.in}|f_j)/|\tilde{f}| \quad (13)$$

where  $|\tilde{f}|$  denotes the number of words in phrase  $\tilde{f}$ .

### 3.3 Adapted Lexical Probability Estimation

Now we briefly describe how to estimate the adapted lexical weight for phrase pairs, which can be adjusted in a similar way to the phrase probability.

Specifically, adopting our method, each word is considered as one phrase consisting of only one word, so

$$\begin{aligned} w(e|f) = & \sum_{t_{f.out}} \sum_{t_{f.in}} w(e|f, t_{f.out}) \\ & \cdot P(t_{f.out}|t_{f.in}) \cdot P(t_{f.in}|f) \end{aligned} \quad (14)$$

Here we obtain  $w(e|f, t_{f.out})$  with a similar approach to  $\phi(\tilde{e}|f, t_{f.out})$ , and calculate  $P(t_{f.out}|t_{f.in})$  and  $P(t_{f.in}|f)$  by resorting to formulas (8) and (9).

With the adjusted lexical translation probability, we resort to formula (4) to update the lexical weight for the phrase pair  $(f, \tilde{e})$ .

## 4 Experiment

We evaluate our method on the Chinese-to-English translation task for the *weblog* text. After a brief description of the experimental setup, we investigate the effects of various factors on the translation system performance.

### 4.1 Experimental setup

In our experiments, the out-of-domain training corpus comes from the FBIS corpus and the Hansards part of LDC2004T07 corpus (54.6K documents with 1M parallel sentences, 25.2M Chinese words and 29M English words). We use the Chinese Sohu weblog in 2009<sup>1</sup> and the English Blog Authorship corpus<sup>2</sup> (Schler et al., 2006) as the in-domain monolingual corpora in the source language and target language, respectively. To obtain more accurate topic information by HTMM, we firstly filter the noisy blog documents and the ones consisting of short sentences. After filtering, there are totally 85K Chinese blog documents with 2.1M sentences and 277K English blog documents with 4.3M sentences used in our experiments. Then, we sample equal numbers of documents from the in-domain monolingual corpora in the source language and the target language to respectively train two in-domain topic models. The web part of the 2006 NIST MT evaluation test data, consisting of 27 documents with 1048 sentences, is used as the development set, and the weblog part of the 2008 NIST MT test data, including 33 documents with 666 sentences, is our test set.

To obtain various topic distributions for the out-of-domain training corpus and the in-domain monolingual corpora in the source language and the target language respectively, we use HTMM tool developed by Gruber et al.(2007) to conduct topic model training. During this process, we empirically set the same parameter values for the HTMM training of different corpora:  $topics = 50$ ,  $\alpha = 1.5$ ,  $\beta = 1.01$ ,  $iters = 100$ . See (Gruber et al., 2007) for the meanings of these parameters. Besides, we set the interpolation weight  $\theta$  in formula (10) to 0.5 by observing the results on development set in the additional experiments.

We choose MOSES, a famous open-source

<sup>1</sup><http://blog.sohu.com/>

<sup>2</sup><http://u.cs.biu.ac.il/~koppel/BlogCorpus.html>

phrase-based machine translation system (Koehn et al., 2007), as the experimental decoder. GIZA++ (Och and Ney, 2003) and the heuristics “grow-diag-final-and” are used to generate a word-aligned corpus, from which we extract bilingual phrases with maximum length 7. We use SRILM Toolkits (Stolcke, 2002) to train two 4-gram language models on the filtered English Blog Authorship corpus and the Xinhua portion of Gigaword corpus, respectively. During decoding, we set the ttable-limit as 20, the stack-size as 100, and perform minimum-error-rate training (Och and Ney, 2003) to tune the feature weights for the log-linear model. The translation quality is evaluated by case-insensitive BLEU-4 metric (Papineni et al., 2002). Finally, we conduct paired bootstrap sampling (Koehn, 2004) to test the significance in BLEU score differences.

## 4.2 Result and Analysis

### 4.2.1 Effect of Different Smoothing Methods

Our first experiments investigate the effect of different smoothing methods for the in-domain phrase-topic distribution: “Noisy-OR” and “Averaging”. We build adapted phrase tables with these two methods, and then respectively use them in place of the out-of-domain phrase table to test the system performance. For the purpose of studying the generality of our approach, we carry out comparative experiments on two sizes of in-domain monolingual corpora: 5K and 40K.

Adaptation Method	(Dev) MT06 Web	(Tst) MT08 Weblog
Baseline	30.98	20.22
Noisy-OR (5K)	31.16	20.45
Averaging (5K)	31.51	20.54
Noisy-OR (40K)	31.87	20.76
Averaging (40K)	31.89	21.11

Table 1: Experimental results using different smoothing methods.

Table 1 reports the BLEU scores of the translation system under various conditions. Using the out-of-domain phrase table, the baseline system achieves a BLEU score of 20.22. In the experiments with the small-scale in-domain monolingual corpora, the

BLEU scores acquired by two methods are 20.45 and 20.54, achieving absolute improvements of 0.23 and 0.32 on the test set, respectively. In the experiments with the large-scale monolingual in-domain corpora, similar results are obtained, with absolute improvements of 0.54 and 0.89 over the baseline system.

From the above experimental results, we know that both “Noisy-OR” and “Averaging” combination methods improve the performance over the baseline, and “Averaging” method seems to be slightly better. This finding fails to echo the promising results in the previous study (Zens and Ney, 2004). This is because the “Noisy-OR” method involves the multiplication of the word-topic distribution (shown in formula (12)), which leads to much sharper phrase-topic distribution than “Averaging” method, and is more likely to introduce bias to the translation probability estimation. Due to this reason, all the following experiments only consider the “Averaging” method.

### 4.2.2 Effect of Combining Two Phrase Tables

In the above experiments, we replace the out-of-domain phrase table with the adapted phrase table. Here we combine these two phrase tables in a log-linear framework to see if we could obtain further improvement. To offer a clear description, we represent the out-of-domain phrase table and the adapted phrase table with “OutBP” and “AdapBP”, respectively.

Used Phrase Table	(Dev) MT06 Web	(Tst) MT08 Weblog
Baseline	30.98	20.22
AdapBp (5K)	31.51	20.54
+ OutBp	31.84	20.70
AdapBp (40K)	31.89	21.11
+ OutBp	32.05	21.20

Table 2: Experimental results using different phrase tables. OutBp: the out-of-domain phrase table. AdapBp: the adapted phrase table.

Table 2 shows the results of experiments using different phrase tables. Applying our adaptation approach, both “AdapBP” and “OutBP + AdapBP” consistently outperform the baseline, and the lat-

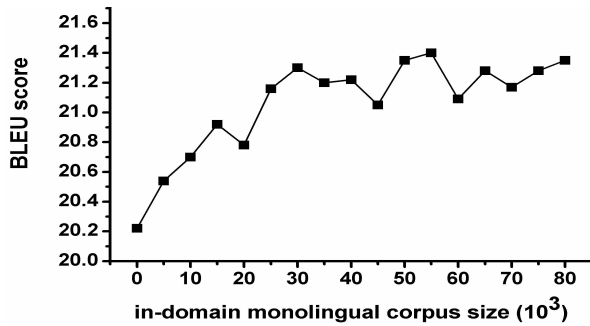


Figure 1: Effect of in-domain monolingual corpus size on translation quality.

ter produces further improvements over the former. Specifically, the BLEU scores of the “OutBP + AdapBP” method are 20.70 and 21.20, which obtain 0.48 and 0.98 points higher than the baseline method, and 0.16 and 0.09 points higher than the ‘AdapBP’ method. The underlying reason is that the probability distribution of each in-domain sentence often converges on some topics in the “AdapBP” method and some translation probabilities are over-estimated, which leads to negative effects on the translation quality. By using two tables together, our approach reduces the bias introduced by “AdapBP”, therefore further improving the translation quality.

#### 4.2.3 Effect of In-domain Monolingual Corpus Size

Finally, we investigate the effect of in-domain monolingual corpus size on translation quality. In the experiment, we try different sizes of in-domain documents to train different monolingual topic models: from 5K to 80K with an increment of 5K each time. Note that here we only focus on the experiments using the “OutBP + AdapBP” method, because this method performs better in the previous experiments.

Figure 1 shows the BLEU scores of the translation system on the test set. It can be seen that the more data, the better translation quality when the corpus size is less than 30K. The overall BLEU scores corresponding to the range of great  $N$  values are generally higher than the ones corresponding to the range of small  $N$  values. For example, the BLEU scores under the condition within the range [25K, 80K] are all higher than the ones within the range [5K, 20K]. When  $N$  is set to 55K, the BLEU

score of our system is **21.40**, with **1.18** gains on the baseline system. This difference is statistically significant at  $P < 0.01$  using the significance test tool developed by Zhang et al.(2004). For this experimental result, we speculate that with the increment of in-domain monolingual data, the corresponding topic models provide more accurate topic information to improve the translation system. However, this effect weakens when the monolingual corpora continue to increase.

## 5 Related work

Most previous researches about translation model adaptation focused on *parallel data collection*. For example, Hildebrand et al.(2005) employed information retrieval technology to gather the bilingual sentences, which are similar to the test set, from available in-domain and out-of-domain training data to build an adaptive translation model. With the same motivation, Munteanu and Marcu (2005) extracted in-domain bilingual sentence pairs from comparable corpora. Since large-scale monolingual corpus is easier to obtain than parallel corpus, there have been some studies on how to generate parallel sentences with monolingual sentences. In this respect, Ueffing et al. (2008) explored semi-supervised learning to obtain synthetic parallel sentences, and Wu et al. (2008) used an in-domain translation dictionary and monolingual corpora to adapt an out-of-domain translation model for the in-domain text.

Differing from the above-mentioned works on the acquirement of bilingual resource, several studies (Foster and Kuhn, 2007; Civera and Juan, 2007; Lv et al., 2007) adopted *mixture modeling* framework to exploit the full potential of the existing parallel corpus. Under this framework, the training corpus is first divided into different parts, each of which is used to train a sub translation model, then these sub models are used together with different weights during decoding. In addition, discriminative weighting methods were proposed to assign appropriate weights to the sentences from training corpus (Matsoukas et al., 2009) or the phrase pairs of phrase table (Foster et al., 2010). Final experimental results show that without using any additional resources, these approaches all improve SMT performance sig-

nificantly.

Our method deals with translation model adaptation by making use of the topical context, so let us take a look at the recent research development on the application of topic models in SMT. Assuming each bilingual sentence constitutes a mixture of hidden topics and each word pair follows a topic-specific bilingual translation model, Zhao and Xing (2006,2007) presented a bilingual topical admixture formalism to improve word alignment by capturing topic sharing at different levels of linguistic granularity. Tam et al.(2007) proposed a bilingual LSA, which enforces one-to-one topic correspondence and enables latent topic distributions to be efficiently transferred across languages, to cross-lingual language modeling and translation lexicon adaptation. Recently, Gong and Zhou (2010) also applied topic modeling into domain adaptation in SMT. Their method employed one additional feature function to capture the topic inherent in the source phrase and help the decoder dynamically choose related target phrases according to the specific topic of the source phrase.

Besides, our approach is also related to context-dependent translation. Recent studies have shown that SMT systems can benefit from the utilization of context information. For example, *trigger-based lexicon model* (Hasan et al., 2008; Mauser et al., 2009) and *context-dependent translation selection* (Chan et al., 2007; Carpuat and Wu, 2007; He et al., 2008; Liu et al., 2008). The former generated triplets to capture long-distance dependencies that go beyond the local context of phrases, and the latter built the classifiers which combine rich context information to better select translation during decoding. With the consideration of various local context features, these approaches all yielded stable improvements on different translation tasks.

As compared to the above-mentioned works, our work has the following differences.

- We focus on how to adapt a translation model for domain-specific translation task with the help of additional in-domain monolingual corpora, which are far from full exploitation in the *parallel data collection* and *mixture modeling* framework.
- In addition to the utilization of in-domain

monolingual corpora, our method is different from the previous works (Zhao and Xing, 2006; Zhao and Xing, 2007; Tam et al., 2007; Gong and Zhou, 2010) in the following aspects: (1) we use a different topic model — HTMM which has different assumption from PLSA and LDA; (2) rather than modeling topic-dependent translation lexicons in the training process, we estimate topic-specific lexical probability by taking account of topical context when extracting word pairs, so our method can also be directly applied to topic-dependent phrase probability modeling. (3) Instead of rescoring phrase pairs online, our approach calculate the translation probabilities offline, which brings no additional burden to translation systems and is suitable to translate the texts without the topic distribution information.

- Different from *trigger-based lexicon model* and *context-dependent translation selection* both of which put emphasis on solving the translation ambiguity by the exploitation of the context information at the sentence level, we adopt the topical context information in our method for the following reasons: (1) the topic information captures the context information beyond the scope of sentence; (2) the topical context information is integrated into the posterior probability distribution, avoiding the sparseness of word or POS features; (3) the topical context information allows for more fine-grained distinction of different translations than the genre information of corpus.

## 6 Conclusion and future work

This paper presents a novel method for SMT system adaptation by making use of the monolingual corpora in new domains. Our approach first estimates the translation probabilities from the out-of-domain bilingual corpus given the topic information, and then rescoring the phrase pairs via topic mapping and phrase-topic distribution probability estimation from in-domain monolingual corpora. Experimental results show that our method achieves better performance than the baseline system, without increasing the burden of the translation system.

In the future, we will verify our method on oth-

er language pairs, for example, Chinese to Japanese. Furthermore, since the in-domain phrase-topic distribution is currently estimated with simple smoothing interpolations, we expect that the translation system could benefit from other sophisticated smoothing methods. Finally, the reasonable estimation of topic number for better translation model adaptation will also become our study emphasis.

## Acknowledgement

The authors were supported by 863 State Key Project (Grant No. 2011AA01A207), National Natural Science Foundation of China (Grant Nos. 61005052 and 61103101), Key Technologies R&D Program of China (Grant No. 2012BAH14F03). We thank the anonymous reviewers for their insightful comments. We are also grateful to Ruiyu Fang and Jinming Hu for their kind help in data processing.

## References

- Michiel Bacchiani and Brian Roark. 2003. Unsupervised Language Model Adaptation. In *Proc. of ICASSP 2003*, pages 224-227.
- Michiel Bacchiani and Brian Roark. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *Computational Linguistics*, pages 477-504.
- Nicola Bertoldi and Marcello Federico. 2009. Domain Adaptation for Statistical Machine Translation with Monolingual Resources. In *Proc. of ACL Workshop 2009*, pages 182-189.
- David M. Blei. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning*, pages 993-1022.
- Ivan Bulyko, Spyros Matsoukas, Richard Schwartz, Long Nguyen and John Makhoul. 2007. Language Model Adaptation in Machine Translation from Speech. In *Proc. of ICASSP 2007*, pages 117-120.
- Marine Carpuat and Dekai Wu. 2007. Improving Statistical Machine Translation Using Word Sense Disambiguation. In *Proc. of EMNLP 2007*, pages 61-72.
- Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2006. Word sense disambiguation improves statistical machine translation. In *Proc. of ACL 2007*, pages 33-40.
- Boxing Chen, George Foster and Roland Kuhn. 2010. Bilingual Sense Similarity for Statistical Machine Translation. In *Proc. of ACL 2010*, pages 834-843.
- David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, pages 201-228.
- David Chiang. 2010. Learning to Translate with Source and Target Syntax. In *Proc. of ACL 2010*, pages 1443-1452.
- Jorge Civera and Alfons Juan. 2007. Domain Adaptation in Statistical Machine Translation with Mixture Modelling. In *Proc. of the Second Workshop on Statistical Machine Translation*, pages 177-180.
- Matthias Eck, Stephan Vogel and Alex Waibel. 2004. Language Model Adaptation for Statistical Machine Translation Based on Information Retrieval. In *Proc. of Fourth International Conference on Language Resources and Evaluation*, pages 327-330.
- Matthias Eck, Stephan Vogel and Alex Waibel. 2005. Low Cost Portability for Statistical Machine Translation Based on N-gram Coverage. In *Proc. of MT Summit 2005*, pages 227-234.
- George Foster and Roland Kuhn. 2007. Mixture Model Adaptation for SMT. In *Proc. of the Second Workshop on Statistical Machine Translation*, pages 128-135.
- George Foster, Cyril Goutte and Roland Kuhn. 2010. Discriminative Instance Weighting for Domain Adaptation in Statistical Machine Translation. In *Proc. of EMNLP 2010*, pages 451-459.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang and Ignacio Thayer. 2006. Scalable Inference and Training of Context-Rich Syntactic Translation Models. In *Proc. of ACL 2006*, pages 961-968.
- Zhengxian Gong and Guodong Zhou. 2010. Improve SMT with Source-side Topic-Document Distributions. In *Proc. of MT SUMMIT 2010*, pages 24-28.
- Amit Gruber, Michal Rosen-Zvi and Yair Weiss. 2007. Hidden Topic Markov Models. In *Journal of Machine Learning Research*, pages 163-170.
- Saša Hasan, Juri Ganitkevitch, Hermann Ney and Jesús Andrés-Ferrer. 2008. Triplet Lexicon Models for Statistical Machine Translation. In *Proc. of EMNLP 2008*, pages 372-381.
- Zhongjun He, Qun Liu and Shouxun Lin. 2008. Improving Statistical Machine Translation using Lexicalized Rule Selection. In *Proc. of COLING 2008*, pages 321-328.
- Almut Silja Hildebrand. 2005. Adaptation of the Translation Model for Statistical Machine Translation based on Information Retrieval. In *Proc. of EAMT 2005*, pages 133-142.
- Thomas Hofmann. 1999. Probabilistic Latent Semantic Indexing. In *Proc. of SIGIR 1999*, pages 50-57.
- Franz Joseph Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, pages 19-51.
- Franz Joseph Och and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, pages 417-449.

- Philipp Koehn, Franz Josef Och and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL 2003*, pages 127-133.
- Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proc. of EMNLP 2004*, pages 388-395.
- 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 *Proc. of ACL 2007, Demonstration Session*, pages 177-180.
- Yang Liu, Qun Liu and Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proc. of ACL 2006*, pages 609-616.
- Yajuan Lv, Jin Huang and Qun Liu. 2007. Improving Statistical Machine Translation Performance by Training Data Selection and Optimization. In *Proc. of EMNLP 2007*, pages 343-350.
- Arne Mauser, Richard Zens and Evgeny Matusov, Saša Hasan and Hermann Ney. 2006. The RWTH Statistical Machine Translation System for the IWSLT 2006 Evaluation. In *Proc. of International Workshop on Spoken Language Translation*, pages 103-110.
- Arne Mauser, Saša Hasan and Hermann Ney. 2009. Extending Statistical Machine Translation with Discriminative and Trigger-Based Lexicon Models. In *Proc. of ACL 2009*, pages 210-218.
- Spyros Matsoukas, Antti-Veikko I. Rosti and Bing Zhang. 2009. Discriminative Corpus Weight Estimation for Machine Translation. In *Proc. of EMNLP 2009*, pages 708-717.
- Nick Ruiz and Marcello Federico. 2011. Topic Adaptation for Lecture Translation through Bilingual Latent Semantic Models. In *Proc. of ACL Workshop 2011*, pages 294-302.
- Kishore Papineni, Salim Roukos, Todd Ward and WeiJing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proc. of ACL 2002*, pages 311-318.
- Jonathan Schler, Moshe Koppel, Shlomo Argamon and James Pennebaker. 2006. Effects of Age and Gender on Blogging. In *Proc. of 2006 AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs*.
- Holger Schwenk and Jean Senellart. 2009. Translation Model Adaptation for an Arabic/french News Translation System by Lightly-supervised Training. In *Proc. of MT Summit XII*.
- Andreas Stolcke. 2002. Srlm - An Extensible Language Modeling Toolkit. In *Proc. of ICSLP 2002*, pages 901-904.
- Yik-Cheung Tam, Ian R. Lane and Tanja Schultz. 2007. Bilingual LSA-based adaptation for statistical machine translation. *Machine Translation*, pages 187-207.
- Nicola Ueffing, Gholamreza Haffari and Anoop Sarkar. 2008. Semi-supervised Model Adaptation for Statistical Machine Translation. *Machine Translation*, pages 77-94.
- Hua Wu, Haifeng Wang and Chengqing Zong. 2008. Domain Adaptation for Statistical Machine Translation with Domain Dictionary and Monolingual Corpora. In *Proc. of COLING 2008*, pages 993-1000.
- Richard Zens and Hermann Ney. 2004. Improvements in phrase-based statistical machine translation. In *Proc. of NAACL 2004*, pages 257-264.
- Ying Zhang, Almut Silja Hildebrand and Stephan Vogel. 2006. Distributed Language Modeling for N-best List Re-ranking. In *Proc. of EMNLP 2006*, pages 216-223.
- Bing Zhao, Matthias Eck and Stephan Vogel. 2004. Language Model Adaptation for Statistical Machine Translation with Structured Query Models. In *Proc. of COLING 2004*, pages 411-417.
- Bing Zhao and Eric P. Xing. 2006. BiTAM: Bilingual Topic AdMixture Models for Word Alignment. In *Proc. of ACL/COLING 2006*, pages 969-976.
- Bing Zhao and Eric P. Xing. 2007. HM-BiTAM: Bilingual Topic Exploration, Word Alignment, and Translation. In *Proc. of NIPS 2007*, pages 1-8.
- Qun Liu, Zhongjun He, Yang Liu and Shouxun Lin. 2008. Maximum Entropy based Rule Selection Model for Syntax-based Statistical Machine Translation. In *Proc. of EMNLP 2008*, pages 89-97.



# A Statistical Model for Unsupervised and Semi-supervised Transliteration Mining

Hassan Sajjad    Alexander Fraser    Helmut Schmid  
Institute for Natural Language Processing  
University of Stuttgart  
{sajjad, fraser, schmid}@ims.uni-stuttgart.de

## Abstract

We propose a novel model to automatically extract transliteration pairs from parallel corpora. Our model is efficient, language pair independent and mines transliteration pairs in a consistent fashion in both unsupervised and semi-supervised settings. We model transliteration mining as an interpolation of transliteration and non-transliteration sub-models. We evaluate on NEWS 2010 shared task data and on parallel corpora with competitive results.

## 1 Introduction

Transliteration mining is the extraction of transliteration pairs from unlabelled data. Most transliteration mining systems are built using labelled training data or using heuristics to extract transliteration pairs. These systems are language pair dependent or require labelled information for training. Our system extracts transliteration pairs in an unsupervised fashion. It is also able to utilize labelled information if available, obtaining improved performance.

We present a novel model of transliteration mining defined as a mixture of a transliteration model and a non-transliteration model. The transliteration model is a joint source channel model (Li et al., 2004). The non-transliteration model assumes no correlation between source and target word characters, and independently generates a source and a target word using two fixed unigram character models. We use Expectation Maximization (EM) to learn parameters maximizing the likelihood of the interpolation of both sub-models. At test time, we label word

pairs as transliterations if they have a higher probability assigned by the transliteration sub-model than by the non-transliteration sub-model.

We extend the unsupervised system to a semi-supervised system by adding a new S-step to the EM algorithm. The S-step takes the probability estimates from unlabelled data (computed in the M-step) and uses them as a backoff distribution to smooth probabilities which were estimated from labelled data. The smoothed probabilities are then used in the next E-step. In this way, the parameters learned by EM are constrained to values which are close to those estimated from the labelled data.

We evaluate our unsupervised and semi-supervised transliteration mining system on the datasets available from the NEWS 2010 shared task on transliteration mining (Kumaran et al., 2010b). We call this task *NEWS10* later on. Compared with a baseline unsupervised system our unsupervised system achieves up to 5% better F-measure. On the NEWS10 dataset, our unsupervised system achieves an F-measure of up to 95.7%, and on three language pairs, it performs better than all systems which participated in NEWS10. We also evaluate our semi-supervised system which additionally uses the NEWS10 labelled data for training. It achieves an improvement of up to 3.7% F-measure over our unsupervised system. Additional experiments on parallel corpora show that we are able to effectively mine transliteration pairs from very noisy data.

The paper is organized as follows. Section 2 describes previous work. Sections 3 and 4 define our unsupervised and semi-supervised models. Section 5 presents the evaluation. Section 6 concludes.

## 2 Previous Work

We first discuss the literature on semi-supervised and supervised techniques for transliteration mining and then describe a previously defined unsupervised system. Supervised and semi-supervised systems use a manually labelled set of training data to learn character mappings between source and target strings. The labelled training data either consists of a few hundred transliteration pairs or of just a few carefully selected transliteration pairs. The NEWS 2010 shared task on transliteration mining (NEWS10) (Kumaran et al., 2010b) is a semi-supervised task conducted on Wikipedia InterLanguage Links (WIL) data. The NEWS10 dataset contains 1000 labelled examples (called the “seed data”) for initial training. All systems which participated in the NEWS10 shared task are either supervised or semi-supervised. They are described in (Kumaran et al., 2010a). Our transliteration mining model can mine transliterations without using any labelled data. However, if there is some labelled data available, our system is able to use it effectively.

The transliteration mining systems evaluated on the NEWS10 dataset generally used heuristic methods, discriminative models or generative models for transliteration mining (Kumaran et al., 2010a).

The heuristic-based system of Jiampojarn et al. (2010) is based on the edit distance method which scores the similarity between source and target words. They presented two discriminative methods – an SVM-based classifier and alignment-based string similarity for transliteration mining. These methods model the conditional probability distribution and require supervised/semi-supervised information for learning. We propose a flexible generative model for transliteration mining usable for both unsupervised and semi-supervised learning.

Previous work on generative approaches uses Hidden Markov Models (Nabende, 2010; Darwish, 2010; Jiampojarn et al., 2010), Finite State Automata (Noeman and Madkour, 2010) and Bayesian learning (Kahki et al., 2011) to learn transliteration pairs from labelled data. Our method is different from theirs as our generative story explains the unlabelled data using a combination of a transliteration and a non-transliteration sub-model. The transliteration model jointly generates source and target

strings, whereas the non-transliteration system generates them independently of each other.

Sajjad et al. (2011) proposed a heuristic-based unsupervised transliteration mining system. We later call it *Sajjad11*. It is the only unsupervised mining system that was evaluated on the NEWS10 dataset up until now, as far as we know. That system is computationally expensive. We show in Section 5 that its runtime is much higher than that of our system.

In this paper, we propose a novel model-based approach to transliteration mining. Our approach is language pair independent – at least for alphabetic languages – and efficient. Unlike the previous unsupervised system, and unlike the supervised and semi-supervised systems we mentioned, our model can be used for both unsupervised and semi-supervised mining in a consistent way.

## 3 Unsupervised Transliteration Mining Model

A source word and its corresponding target word can be character-aligned in many ways. We refer to a possible alignment sequence which aligns a source word  $e$  and a target word  $f$  as “ $a$ ”. The function  $Align(e, f)$  returns the set of all valid alignment sequences  $a$  of a word pair  $(e, f)$ . The joint transliteration probability  $p_1(e, f)$  of a word pair is the sum of the probabilities of all alignment sequences:

$$p_1(e, f) = \sum_{a \in Align(e, f)} p(a) \quad (1)$$

Transliteration systems are trained on a list of transliteration pairs. The alignment between the transliteration pairs is learned with Expectation Maximization (EM). We use a simple unigram model, so an alignment sequence from function  $Align(e, f)$  is a combination of 0–1, 1–1, and 1–0 character alignments between a source word  $e$  and its transliteration  $f$ . We refer to a character alignment unit as “*multigram*” later on and represent it by the symbol “ $q$ ”. A sequence of multigrams forms an alignment of a source and target word. The probability of a sequence of multigrams  $a$  is the product of the probabilities of the multigrams it contains.

$$p(a) = p(q_1, q_2, \dots, q_{|a|}) = \prod_{j=1}^{|a|} p(q_j) \quad (2)$$

While transliteration systems are trained on a clean list of transliteration pairs, our transliteration mining system has to learn from data containing both transliterations and non-transliterations. The transliteration model  $p_1(e, f)$  handles only the transliteration pairs. We propose a second model  $p_2(e, f)$  to deal with non-transliteration pairs (the “*non-transliteration model*”). Interpolation with the non-transliteration model allows the transliteration model to concentrate on modelling transliterations during EM training. After EM training, transliteration word pairs are assigned a high probability by the transliteration submodel and a low probability by the non-transliteration submodel, and vice versa for non-transliteration pairs. This property is exploited to identify transliterations.

In a non-transliteration word pair, the characters of the source and target words are unrelated. We model them as randomly seeing a source word and a target word together. The non-transliteration model uses random generation of characters from two unigram models. It is defined as follows:

$$p_2(e, f) = p_E(e) p_F(f) \quad (3)$$

$$p_E(e) = \prod_{i=1}^{|e|} p_E(e_i) \text{ and } p_F(f) = \prod_{i=1}^{|f|} p_F(f_i).$$

The transliteration mining model is an interpolation of the transliteration model  $p_1(e, f)$  and the non-transliteration model  $p_2(e, f)$ :

$$p(e, f) = (1 - \lambda)p_1(e, f) + \lambda p_2(e, f) \quad (4)$$

$\lambda$  is the prior probability of non-transliteration.

### 3.1 Model Estimation

In this section, we discuss the estimation of the parameters of the transliteration model  $p_1(e, f)$  and the non-transliteration model  $p_2(e, f)$ .

The non-transliteration model consists of two unigram character models. Their parameters are estimated from the source and target words of the training data, respectively, and the parameters do not change during EM training.

For the transliteration model, we implement a simplified form of the grapheme-to-phoneme converter, g2p (Bisani and Ney, 2008). In the following, we use notations from Bisani and Ney (2008). g2p learns m-to-n character alignments between a source and a target word. We restrict ourselves to 0–1, 1–1, 1–0 character alignments and to a unigram

model.<sup>1</sup> The Expectation Maximization (EM) algorithm is used to train the model. It maximizes the likelihood of the training data. In the E-step the EM algorithm computes expected counts for the multigrams and in the M-step the multigram probabilities are reestimated from these counts. These two steps are iterated. For the first EM iteration, the multigram probabilities are initialized with a uniform distribution and  $\lambda$  is set to 0.5.

The expected count of a multigram  $q$  (E-step) is computed by multiplying the posterior probability of each alignment  $a$  with the frequency of  $q$  in  $a$  and summing these weighted frequencies over all alignments of all word pairs.

$$c(q) = \sum_{i=1}^N \sum_{a \in \text{Align}(e_i, f_i)} \frac{(1 - \lambda)p_1(a, e_i, f_i)}{p(e_i, f_i)} n_q(a)$$

$n_q(a)$  is here the number of times the multigram  $q$  occurs in the sequence  $a$  and  $p(e_i, f_i)$  is defined in Equation 4. The new estimate of the probability of a multigram is given by:

$$p(q) = \frac{c(q)}{\sum_{q'} c(q')} \quad (5)$$

Likewise, we calculate the expected count of non-transliterations by summing the posterior probabilities of non-transliteration given each word pair:

$$c_{ntr} = \sum_{i=1}^N p_{ntr}(e_i, f_i) = \sum_{i=1}^N \frac{\lambda p_2(e_i, f_i)}{p(e_i, f_i)} \quad (6)$$

$\lambda$  is then reestimated by dividing the expected count of non-transliterations by  $N$ .

### 3.2 Implementation Details

We use the Forward-Backward algorithm to estimate the counts of multigrams. The algorithm has a forward variable  $\alpha$  and a backward variable  $\beta$  which are calculated in the standard way (Deligne and Bimbot, 1995). Consider a node  $r$  which is connected with a node  $s$  via an arc labelled with the multigram  $q$ . The expected count of a transition between  $r$  and  $s$  is calculated using the forward and backward probabilities as follows:

$$\gamma'_{rs} = \frac{\alpha(r) p(q) \beta(s)}{\alpha(E)} \quad (7)$$

<sup>1</sup>In preliminary experiments, using an n-gram order of greater than one or more than one character on the source side or the target side or both sides of the multigram caused the transliteration model to incorrectly learn non-transliteration information from the training data.

where  $E$  is the final node of the graph.

We multiply the expected count of a transition by the posterior probability of transliteration ( $1 - p_{ntr}(e, f)$ ) which indicates how likely the string pair is to be a transliteration. The counts  $\gamma_{rs}$  are then summed for all multigram types  $q$  over all training pairs to obtain the frequencies  $c(q)$  which are used to reestimate the multigram probabilities according to Equation 5.

## 4 Semi-supervised Transliteration Mining Model

Our unsupervised transliteration mining system can be applied to language pairs for which no labelled data is available. However, the unsupervised system is focused on high recall and also mines close transliterations (see Section 5 for details). In a task dependent scenario, it is difficult for the unsupervised system to mine transliteration pairs according to the details of a particular definition of what is considered a transliteration (which may vary somewhat with the task). In this section, we propose an extension of our unsupervised model which overcomes this shortcoming by using labelled data. The idea is to rely on probabilities from labelled data where they can be estimated reliably and to use probabilities from unlabelled data where the labelled data is sparse. This is achieved by smoothing the labelled data probabilities using the unlabelled data probabilities as a backoff.

### 4.1 Model Estimation

We calculate the unlabelled data probabilities in the E-step using Equation 4. For labelled data (containing only transliterations) we set  $\lambda = 0$  and get:

$$p(e, f) = \sum_{a \in \text{Align}(e, f)} p_1(e, f, a) \quad (8)$$

In every EM iteration, we smooth the probability distribution in such a way that the estimates of the multigrams of the unlabelled data that do not occur in the labelled data would be penalized. We obtain this effect by smoothing the probability distribution of unlabelled and labelled data using a technique similar to Witten-Bell smoothing (Witten and Bell, 1991), as we describe below.

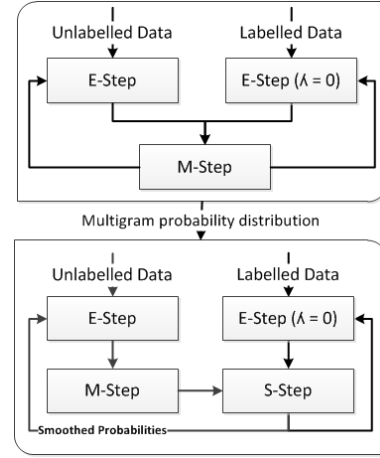


Figure 1: Semi-supervised training

### 4.2 Implementation Details

We divide the training process of semi-supervised mining in two steps as shown in Figure 1. The first step creates a reasonable alignment of the labelled data from the unlabelled data from which multigram counts can be obtained. The labelled data is a small list of transliteration pairs. Therefore we use the unlabelled data to help correctly align it and train our unsupervised mining system on the combined labelled and unlabelled training data. In the expectation step, the prior probability of non-transliteration  $\lambda$  is set to zero on the labelled data since it contains only transliterations. The first step passes the resulting multigram probability distribution to the second step.

We start the second step with the probability estimates from the first step and run the E-step separately on labelled and unlabelled data. The E-step on the labelled data is done using Equation 8, which forces the posterior probability of non-transliteration to zero, while the E-step on the unlabelled data uses Equation 4. After the two E-steps, we estimate a probability distribution from the counts obtained from the unlabelled data (M-step) and use it as a backoff distribution in computing smoothed probabilities from the labelled data counts (S-step).

The smoothed probability estimate  $\hat{p}(q)$  is:

$$\hat{p}(q) = \frac{c_s(q) + \eta_s p(q)}{N_s + \eta_s} \quad (9)$$

where  $c_s(q)$  is the labelled data count of the multigram  $q$ ,  $p(q)$  is the unlabelled data probability estimate, and  $N_s = \sum_q c_s(q)$ , and  $\eta_s$  is the number of different multigram types observed in the Viterbi alignment of the labelled data.

## 5 Evaluation

We evaluate our unsupervised system and semi-supervised system on two tasks, NEWS10 and parallel corpora. NEWS10 is a standard task on transliteration mining from WIL. On NEWS10, we compare our results with the unsupervised mining system of Sajjad et al. (2011), the best supervised and semi-supervised systems presented at NEWS10 (Kumaran et al., 2010b) and the best supervised and semi-supervised results reported in the literature for the NEWS10 task. For the challenging task of mining from parallel corpora, we use the English/Hindi and English/Arabic gold standard provided by Sajjad et al. (2011) to evaluate our results.

### 5.1 Experiments using the NEWS10 Dataset

We conduct experiments on four language pairs: English/Arabic, English/Hindi, English/Tamil and English/Russian using data provided at NEWS10. Every dataset contains training data, seed data and reference data. The NEWS10 data consists of pairs of titles of the same Wikipedia pages written in different languages, which may be transliterations or translations. The seed data is a list of 1000 transliteration pairs provided to semi-supervised systems for initial training. We use the seed data only in our semi-supervised system, and not in the unsupervised system. The reference data is a small subset of the training data which is manually annotated with positive and negative examples.

#### 5.1.1 Training

We word-aligned the parallel phrases of the training data using GIZA++ (Och and Ney, 2003), and symmetrized the alignments using the grow-diag-final-and heuristic (Koehn et al., 2003). We extract all word pairs which occur as 1-to-1 alignments (like Sajjad et al. (2011)) and later refer to them as the *word-aligned list*. We compared the word-aligned list with the NEWS10 reference data and found that the word-aligned list is missing some transliteration pairs because of word-alignment errors. We built another list by adding a word pair for every source word that cooccurs with a target word in a parallel phrase/sentence and call it the *cross-product list* later on. The cross-product list is noisier but contains almost all transliteration pairs in the corpus.

	Word-aligned			Cross-product		
	P	R	F	P	R	F
<b>EA</b>	27.8	97.1	43.3	14.3	98.0	25.0
<b>EH</b>	42.5	98.7	59.4	20.5	99.6	34.1
<b>ET</b>	32.0	98.1	48.3	17.2	99.6	29.3
<b>ER</b>	25.5	95.6	40.3	12.8	99.0	22.7

Table 1: Statistics of word-aligned and cross-product list calculated from the NEWS10 dataset, before mining. *EA* is English/Arabic, *EH* is English/Hindi, *ET* is English/Tamil and *ER* is English/Russian

Table 1 shows the statistics of the word-aligned list and the cross-product list calculated using the NEWS10 reference data.<sup>2</sup> The word-aligned list calculated from the NEWS10 dataset is used to compare our unsupervised system with the unsupervised system of Sajjad et al. (2011) on the same training data. All the other experiments on NEWS10 use cross-product lists. We remove numbers from both lists as they are defined as non-transliterations (Kumaran et al., 2010b).

#### 5.1.2 Unsupervised Transliteration Mining

We run our unsupervised transliteration mining system on the word-aligned list and the cross-product list. The word pairs with a posterior probability of transliteration  $1 - p_{ntr}(e, f) = 1 - \lambda p_2(e_i, f_i) / p(e_i, f_i)$  greater than 0.5 are selected as transliteration pairs.

We compare our unsupervised system with the unsupervised system of Sajjad11. Our unsupervised system trained on the word-aligned list shows F-measures of 91.7%, 95.5%, 92.9% and 77.7% which is 4.3%, 3.3%, 2.8% and 1.7% better than the system of Sajjad11 on English/Arabic, English/Hindi, English/Tamil and English/Russian respectively.

Sajjad11 is computationally expensive. For instance, a phrase-based statistical MT system is built once in every iteration of the heuristic procedure. We ran Sajjad11 on the English/Russian word-aligned list using a 2.4 GHz Dual-Core AMD machine, which took almost 10 days. On the same machine, our transliteration mining system only takes 1.5 hours to finish the same experiment.

<sup>2</sup>Due to inconsistent word definition used in the reference data, we did not achieve 100% recall in our cross-product list. For example, the underscore is defined as a word boundary for English WIL phrases. This assumption is not followed for certain phrases like "New\_York" and "New\_Mexico".

	Unsupervised		Semi-supervised/Supervised			
	<i>SJD</i>	<i>O<sub>U</sub></i>	<i>O<sub>S</sub></i>	<i>S<sub>Best</sub></i>	<i>GR</i>	<i>DBN</i>
<b>EA</b>	87.4	92.4	92.7	91.5	94.1	-
<b>EH</b>	92.2	95.7	96.3	94.4	93.2	95.5
<b>ET</b>	90.1	93.2	94.6	91.4	95.5	93.9
<b>ER</b>	76.0	79.4	83.1	87.5	92.3	82.5

Table 2: F-measure results on NEWS10 datasets where *SJD* is the unsupervised system of Sajjad11, *O<sub>U</sub>* is our unsupervised system built on the cross-product list, *O<sub>S</sub>* is our semi-supervised system, *S<sub>Best</sub>* is the best NEWS10 system, *GR* is the supervised system of Kahki et al. (2011) and *DBN* is the semi-supervised system of Nabende (2011)

Our unsupervised mining system built on the cross-product list consistently outperforms the one built on the word-aligned list. Later, we consider only the system built on the cross-product list. Table 2 shows the results of our unsupervised system *O<sub>U</sub>* in comparison with the unsupervised system of Sajjad11 (*SJD*), the best semi-supervised systems presented at NEWS10 (*S<sub>BEST</sub>*) and the best semi-supervised results reported on the NEWS10 dataset (*GR*, *DBN*). On three language pairs, our unsupervised system performs better than all semi-supervised systems which participated in NEWS10. It has competitive results with the best supervised results reported on NEWS10 datasets. On English/Hindi, our unsupervised system outperforms the state-of-the-art supervised and semi-supervised systems. Kahki et al. (2011) (*GR*) achieved the best results on English/Arabic, English/Tamil and English/Russian. For the English/Arabic task, they normalized the data using language dependent heuristics<sup>3</sup> and also used a non-standard evaluation method (discussed in Section 5.1.4).

On the English/Russian dataset, our unsupervised system faces the problem that it extracts cognates as transliterations. The same problem was reported in Sajjad et al. (2011). Cognates are close transliterations which differ by only one or two characters from an exact transliteration pair. The unsupervised system learns to delete the additional one or two characters with a high probability and incorrectly mines such word pairs as transliterations.

<sup>3</sup>They applied an Arabic word segmenter which uses language dependent information. Arabic long vowels which have identical sound but are written differently were merged to one form. English characters were normalized by dropping accents.

	Unsupervised			Semi-supervised		
	P	R	F	P	R	F
<b>EA</b>	89.2	95.7	92.4	92.9	92.4	92.7
<b>EH</b>	92.6	99.0	95.7	95.5	97.0	96.3
<b>ET</b>	88.3	98.6	93.2	93.4	95.8	94.6
<b>ER</b>	67.2	97.1	79.4	74.0	94.9	83.1

Table 3: Precision(P), Recall(R) and F-measure(F) of our unsupervised and semi-supervised transliteration mining systems on NEWS10 datasets

### 5.1.3 Semi-supervised Transliteration Mining

Our semi-supervised system uses similar initialization of the parameters as used for unsupervised system. Table 2 shows on three language pairs, our semi-supervised system *O<sub>S</sub>* only achieves a small gain in F-measure over our unsupervised system *O<sub>U</sub>*. This shows that the unlabelled training data is already providing most of the transliteration information. The seed data is used to help the transliteration mining system to learn the right definition of transliteration. On the English/Russian dataset, our semi-supervised system achieves almost 7% increase in precision with a 2.2% drop in recall compared to our unsupervised system. This provides a 3.7% gain on F-measure. The increase in precision shows that the seed data is helping the system in disambiguating transliteration pairs from cognates.

### 5.1.4 Discussion

The unsupervised system produces lists with high recall. The semi-supervised system tends to better balance out precision and recall. Table 3 compares the precision, recall and F-measure of our unsupervised and semi-supervised mining systems.

The errors made by our semi-supervised system can be classified into the following categories:

**Pronunciation differences:** English proper names may be pronounced differently in other languages. Sometimes, English short vowels are converted to long vowels in Hindi such as the English word “Lanthanum” which is pronounced “Laanthanum” in Hindi. Our transliteration mining system wrongly extracts such pairs as transliterations.

In some cases, different vowels are used in two languages. The English word “January” is pronounced as “Janvary” in Hindi. Such word pairs are non-transliterations according to the gold standard but our system extracts them as transliterations. Ta-

English	Hindi	English	Hindi
Lanthanum	लाञ्छनम्/Laanthanum	Sailendra	शैलेन्द्र/Shalendra
January	जनवरी/Janvary	August	अगस्त/Aagast

Table 4: Word pairs with pronunciation differences

English	Arabic	English	Arabic
Basrah	البصرة/Albasrah	Nasr	النصر/Alnasr
Kuwait	الكويت/Alkuwait	Riyadh	الرياض/Alriyadh

Table 5: Examples of word pairs which are wrongly annotated as transliterations in the gold standard

ble 4 shows a few examples of such word pairs.

**Inconsistencies in the gold standard:** There are several inconsistencies in the gold standard where our transliteration system correctly identifies a word pair as a transliteration but it is marked as a non-transliteration or vice versa. Consider the example of the English word “George” which is pronounced as “Jaarj” in Hindi. Our semi-supervised system learns this as a non-transliteration but it is wrongly annotated as a transliteration in the gold standard.

Arabic nouns have an article “al” attached to them which is translated in English as “the”. There are various cases in the training data where an English noun such as “Quran” is matched with an Arabic noun “alQuran”. Our mining system classifies such cases as non-transliterations, but 24 of them are incorrectly annotated as transliterations in the gold standard. We did not correct this, and are therefore penalized. Kahki et al. (2011) preprocessed such Arabic words and separated “al” from the noun “Quran” before mining. They report a match if the version of the Arabic word with “al” appears with the corresponding English word in the gold standard. Table 5 shows examples of word pairs which are wrongly annotated as transliterations.

**Cognates:** Sometimes a word pair differs by only one or two ending characters from a true transliteration. For example in the English/Russian training data, the Russian nouns are marked with cases whereas their English counterparts do not mark the case or translate it as a separate word. Often the Russian word differs only by the last character from a correct transliteration of the English word. Due to the large amount of such word pairs in the English/Russian data, our mining system learns to delete the final case marking characters from the Russian words. It assigns a high transliteration prob-

English	Russian	English	Russian
Studio	Студия/Studiya	Catalonia	Каталонии/Katalonii
Estonia	Эстонии/Estonii	Geography	География/Geografiya

Table 6: A few examples of English/Russian cognates

ability to these word pairs and extracts them as transliterations. Table 6 shows some examples.

There are two English/Russian supervised systems which are better than our semi-supervised system. The Kahki et al. (2011) system is built on seed data only. Jiampojarn et al. (2010)’s best system on English/Russian is based on the edit distance method. Both of these systems are focused on high precision. Our semi-supervised system is focused on high recall at the cost of lower precision.<sup>4</sup>

## 5.2 Transliteration Mining using Parallel Corpora

The percentage of transliteration pairs in the NEWS10 datasets is high. We further check the effectiveness of our unsupervised and semi-supervised mining systems by evaluating them on parallel corpora with as few as 2% transliteration pairs.

We conduct experiments using two language pairs, English/Hindi and English/Arabic. The English/Hindi corpus is from the shared task on word alignment organized as part of the ACL 2005 Workshop on Building and Using Parallel Texts (WA05) (Martin et al., 2005). For English/Arabic, we use 200,000 parallel sentences from the United Nations (UN) corpus (Eisele and Chen, 2010). The English/Hindi and English/Arabic transliteration gold standards were provided by Sajjad et al. (2011).

### 5.2.1 Experiments

We follow the procedure for creating the training data described in Section 5.1.1 and build a word-aligned list and a cross-product list from the parallel corpus. We first train and test our unsupervised mining system on the word-aligned list and compare our results with Sajjad et al. Table 7 shows the results. Our unsupervised system achieves 0.6% and 1.8% higher F-measure than Sajjad et al. respectively.

The cross-product list is huge in comparison to the word-aligned list. It is noisier than the word-

<sup>4</sup>We implemented a bigram version of our system to learn the contextual information at the end of the word pairs, but only achieved a gain of less than 1% F-measure over our unigram semi-supervised system. Details are omitted due to space.

	TP	FN	TN	FP	P	R	F
<i>EH<sub>SJD</sub></i>	170	10	2039	45	79.1	94.4	86.1
<i>EH<sub>O</sub></i>	176	4	2034	50	77.9	97.8	86.7
<i>EA<sub>SJD</sub></i>	197	91	6580	59	77.0	68.4	72.5
<i>EA<sub>O</sub></i>	288	0	6440	199	59.1	100	74.3

Table 7: Transliteration mining results of our unsupervised system and Sajjad11 system trained and tested on the word-aligned list of English/Hindi and English/Arabic parallel corpus

	TP	FN	TN	FP	P	R	F
<i>EH<sub>U</sub></i>	393	19	12279	129	75.3	95.4	84.2
<i>EH<sub>S</sub></i>	365	47	12340	68	84.3	88.6	86.4
<i>EA<sub>U</sub></i>	277	11	6444	195	58.7	96.2	72.9
<i>EA<sub>S</sub></i>	272	16	6497	142	65.7	94.4	77.5

Table 8: Transliteration mining results of our unsupervised and semi-supervised systems trained on the word-aligned list and tested on the cross-product list of English/Hindi and English/Arabic parallel corpus

aligned list but has almost 100% recall of transliteration pairs. The English-Hindi cross-product list has almost 55% more transliteration pairs (412 types) than the word-aligned list (180 types). We can not report these numbers on the English/Arabic cross-product list since the English/Arabic gold standard is built on the word-aligned list.

In order to keep the experiment computationally inexpensive, we train our mining systems on the word-aligned list and test them on the cross-product list.<sup>5</sup> We also perform the first semi-supervised evaluation on this task. For our semi-supervised system, we additionally use the English/Hindi and English/Arabic seed data provided by NEWS10.

Table 8 shows the results of our unsupervised and semi-supervised systems on the English/Hindi and English/Arabic parallel corpora. Our unsupervised system achieves higher recall than our semi-supervised system but lower precision. The semi-supervised system shows an improvement in F-measure for both language pairs. We looked into the errors made by our systems. The mined transliteration pairs of our unsupervised system contains 65 and 111 close transliterations for the English/Hindi and English/Arabic task respectively.

<sup>5</sup>There are some multigrams of the cross-product list which are unknown to the model learned on the word-aligned list. We define their probability as the inverse of the number of multigram tokens in the Viterbi alignment of the labelled and unlabelled data together.

The close transliterations only differ by one or two characters from correct transliterations. We think these pairs provide transliteration information to the systems and help them to avoid problems with data sparseness. Our semi-supervised system uses the seed data to identify close transliterations as non-transliterations and decreases the number of false positives. They are reduced to 35 and 89 for English/Hindi and English/Arabic respectively. The seed data and the training data used in the semi-supervised system are from different domains (Wikipedia and UN). Seed data extracted from the same domain is likely to work better, resulting in even higher scores than we have reported.

## 6 Conclusion and Future Work

We presented a novel model to automatically mine transliteration pairs. Our approach is efficient and language pair independent (for alphabetic languages). Both the unsupervised and semi-supervised systems achieve higher accuracy than the only unsupervised transliteration mining system we are aware of and are competitive with the state-of-the-art supervised and semi-supervised systems. Our semi-supervised system outperformed our unsupervised system, in particular in the presence of prevalent cognates in the Russian/English data.

In future work, we plan to adapt our approach to language pairs where one language is alphabetic and the other language is non-alphabetic such as English/Japanese. These language pairs require one-to-many character mappings to learn transliteration units, while our current system only learns unigram character alignments.

## Acknowledgments

The authors wish to thank the anonymous reviewers. We would like to thank Syed Aoun Raza for discussions of implementation efficiency. Hassan Sajjad was funded by the Higher Education Commission of Pakistan. Alexander Fraser was funded by Deutsche Forschungsgemeinschaft grant Models of Morphosyntax for Statistical Machine Translation. Helmut Schmid was supported by Deutsche Forschungsgemeinschaft grant SFB 732. This work was supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886. This publication only reflects the authors' views.



## References

- Maximilian Bisani and Hermann Ney. 2008. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, 50(5).
- Kareem Darwish. 2010. Transliteration mining with phonetic conflation and iterative training. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.
- Sabine Deligne and Frédéric Bimbot. 1995. Language modeling by variable length sequences : Theoretical formulation and evaluation of multigrams. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, Los Alamitos, CA, USA.
- Andreas Eisele and Yu Chen. 2010. MultiUN: A multilingual corpus from United Nation documents. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta.
- Sittichai Jiampojarnam, Kenneth Dwyer, Shane Bergsma, Aditya Bhargava, Qing Dou, Mi-Young Kim, and Grzegorz Kondrak. 2010. Transliteration generation and mining with limited training resources. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.
- Ali El Kahki, Kareem Darwish, Ahmed Saad El Din, Mohamed Abd El-Wahab, Ahmed Hefny, and Waleed Ammar. 2011. Improved transliteration mining using graph reinforcement. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Edinburgh, UK.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference*, Edmonton, Canada.
- A Kumaran, Mitesh M. Khapra, and Haizhou Li. 2010a. Report of NEWS 2010 transliteration mining shared task. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.
- A Kumaran, Mitesh M. Khapra, and Haizhou Li. 2010b. Whitepaper of NEWS 2010 shared task on transliteration mining. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.
- Haizhou Li, Zhang Min, and Su Jian. 2004. A joint source-channel model for machine transliteration. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Barcelona, Spain.
- Joel Martin, Rada Mihalcea, and Ted Pedersen. 2005. Word alignment for languages with scarce resources. In *ParaText '05: Proceedings of the ACL Workshop on Building and Using Parallel Texts*, Morristown, NJ, USA.
- Peter Nabende. 2010. Mining transliterations from wikipedia using pair hmms. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.
- Peter Nabende. 2011. Mining transliterations from Wikipedia using dynamic bayesian networks. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, Hissar, Bulgaria.
- Sara Noeman and Amgad Madkour. 2010. Language independent transliteration mining system using finite state automata framework. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.
- Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1).
- Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2011. An algorithm for unsupervised transliteration mining with an application to word alignment. In *Proceedings of the 49th Annual Conference of the Association for Computational Linguistics*, Portland, USA.
- Ian H. Witten and Timothy C. Bell. 1991. The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. In *IEEE Transactions on Information Theory*, volume 37.

# Modified Distortion Matrices for Phrase-Based Statistical Machine Translation

Arianna Bisazza and Marcello Federico

Fondazione Bruno Kessler

Trento, Italy

{bisazza, federico}@fbk.eu

## Abstract

This paper presents a novel method to suggest long word reorderings to a phrase-based SMT decoder. We address language pairs where long reordering concentrates on few patterns, and use fuzzy chunk-based rules to predict likely reorderings for these phenomena. Then we use reordered  $n$ -gram LMs to rank the resulting permutations and select the  $n$ -best for translation. Finally we encode these reorderings by modifying selected entries of the distortion cost matrix, on a per-sentence basis. In this way, we expand the search space by a much finer degree than if we simply raised the distortion limit. The proposed techniques are tested on Arabic-English and German-English using well-known SMT benchmarks.

## 1 Introduction

Despite the large research effort devoted to the modeling of word reordering, this remains one of the main obstacles to the development of accurate SMT systems for many language pairs. On one hand, the phrase-based approach (PSMT) (Och, 2002; Zens et al., 2002; Koehn et al., 2003), with its shallow and loose modeling of linguistic equivalences, appears as the most competitive choice for closely related language pairs with similar clause structures, both in terms of quality and of efficiency. On the other, tree-based approaches (Wu, 1997; Yamada, 2002; Chiang, 2005) gain advantage, at the cost of higher complexity and isomorphism assumptions, on language pairs with radically different word orders.

Lying between these two extremes are language pairs where most of the reordering happens locally,

and where long reorderings can be isolated and described by a handful of linguistic rules. Notable examples are the family-unrelated Arabic-English and the related German-English language pairs. Interestingly, on these pairs, PSMT generally prevails over tree-based SMT<sup>1</sup>, producing overall high-quality outputs and isolated but critical reordering errors that undermine the global sentence meaning.

Previous works on this type of language pairs have mostly focused on source reordering prior to translation (Xia and McCord, 2004; Collins et al., 2005), or on sophisticated reordering models integrated into decoding (Koehn et al., 2005; Al-Onaizan and Papineni, 2006), achieving mixed results. To merge the best of both approaches – namely, access to rich context in the former and natural coupling of reordering and translation decisions in the latter – we introduce *modified distortion matrices*: a novel method to seamlessly provide to the decoder a set of likely long reorderings pre-computed for a given input sentence. Added to the usual space of local permutations defined by a low distortion limit (DL), this results in a linguistically informed definition of the search space that simplifies the task of the in-decoder reordering model, besides decreasing its complexity.

The paper is organized as follows. After reviewing a selection of relevant works, we analyze salient reordering patterns in Arabic-English and German-English, and describe the corresponding chunk-based reordering rule sets. In the following sections we present a reordering selection technique based on

---

<sup>1</sup>A good comparison of phrase-based and tree-based approaches across language pairs with different reordering levels can be found in (Zollmann et al., 2008).

reordered n-gram LMs and, finally, explain the notion of modified distortion matrices. In the last part of the paper, we evaluate the proposed techniques on two popular MT tasks.

## 2 Previous work

Pre-processing approaches to word reordering aim at permuting input words in a way that minimizes the reordering needed for translation: *deterministic reordering* aims at finding a single optimal reordering for each input sentence, which is then translated monotonically (Xia and McCord, 2004) or with a low DL (Collins et al., 2005; Habash, 2007); *non-deterministic reordering* encodes multiple alternative reorderings into a word lattice and lets a monotonic decoder find the best path according to its models (Zhang et al., 2007; Crego and Habash, 2008; Elming and Habash, 2009; Niehues and Kolss, 2009). The latter approaches are ideally conceived as alternative to in-decoding reordering, and therefore require an exhaustive reordering rule set. Two recent works (Bisazza and Federico, 2010; Andreas et al., 2011) opt instead for a *hybrid way*: rules are used to generate multiple likely reorderings, but only for a specific phenomenon – namely verb-initial clauses in Arabic. This yields sparse reordering lattices that can be translated with a regular decoder performing additional reordering.

Reordering rules for pre-processing are either manually written (Collins et al., 2005) or automatically learned from syntactic parses (Xia and McCord, 2004; Habash, 2007; Elming and Habash, 2009), shallow syntax chunks (Zhang et al., 2007; Crego and Habash, 2008) or part-of-speech labels (Niehues and Kolss, 2009). Similarly to hybrid approaches, in this work we use few linguistically informed rules to generate multiple reorderings for selected phenomena but, as a difference, we do not employ lattices to represent them. We also include a competitive in-decoding reordering model in all the systems used to evaluate our methods.

Another large body of work is devoted to the modeling of reordering decisions inside decoding, based on a decomposition of the problem into a sequence of basic reordering steps. Existing approaches range from basic linear distortion to more complex models that are conditioned on the words being translated.

The *linear distortion model* (Koehn et al., 2003) encourages monotonic translations by penalizing source position jumps proportionally to their length. If used alone, this model is inadequate for language pairs with different word orders. Green et al. (2010) tried to improve it with a future distortion cost estimate. Thus they were able to preserve baseline performance at a very high DL, but not to improve it. *Lexicalized phrase orientation models* (Tillmann, 2004; Koehn et al., 2005; Zens and Ney, 2006; Galley and Manning, 2008) predict the orientation of a phrase with respect to the last translated one. These models are known to well handle local reordering and are widely adopted by the PSMT community. However, they are unsuitable to model long reordering as they classify as “discontinuous” every phrase that does not immediately follow or precede the last translated one. *Lexicalized distortion models* predict the jump from the last translated word to the next one, with a class for each possible jump length (Al-Onaizan and Papineni, 2006), or bin of lengths (Green et al., 2010). These models are conceived to deal with long reordering, but can easily suffer from data sparseness, especially for longer jumps occurring less frequently.

Following a typical sequence modeling approach, Feng et al. (2010) train n-gram language models on source data previously reordered in accordance to the target language translation. This method does not directly model reordering decisions, but rather word sequences produced by them. Despite their high perplexities, reordered LMs yield some improvements when integrated to a PSMT baseline that already includes a discriminative phrase orientation model (Zens and Ney, 2006). In this work we use similar models to rank sets of chunk permutations.

Attempting to improve the reordering space definition, Yahyaei and Monz (2010) train a classifier to guess the most likely jump length at each source position, then use its predictions to dynamically set the DL. Translation improvements are obtained on a simple task with mostly short sentences (BTEC).

Modifying the distortion function, as proposed in this paper, makes it possible to expand the permutation search space by a much finer degree than varying the DL does.

### 3 Long reordering patterns

Our study focuses on Arabic-English and German-English: two language pairs characterized by uneven distributions of word-reordering phenomena, with long-range movements concentrating on few patterns. In **Arabic-English**, the internal order of most noun phrases needs to be reversed during translation, which is generally well handled by phrase-internal reordering or local distortion. At the constituent level, instead, Arabic admits both SV(O) and VS(O) orders, the latter causing problematic long reorderings. Common errors due to this issue are the absence of main verb in the English translation, or the placement of the main verb before its own subject. In both cases, adequacy is seriously compromised. In **German-English**, the noun phrase structure is similar between source and target languages. However, at the constituent level, the *verb-second* order of German main clauses conflicts with the rigid SVO structure of English, as does the clause-final verb position of German subordinate clauses. As a further complication, German compound verbs are split apart so that the non-finite element (main verb) can appear long after the inflected auxiliary or modal.

Thanks to sophisticated reordering models, state-of-the-art PSMT systems are generally good at handling *local* reordering phenomena that are not captured by phrase-internal reordering. However, they typically fail to predict long reorderings. We believe this is mainly not the fault of the reordering models, but rather of a *too coarse definition* of the search space. To have a concrete idea, consider that a small change of the DL from 5 to 6 words, in a sentence of 8, makes the number of explorable permutations increase from about 9,000 to 22,000. Existing models cannot be powerful enough to deal with such a rapidly growing search space.

As a result, decoding at very high DLs is not a good solution for these language pairs. Indeed, decent performances are obtained within a low or medium DL, but this obviously comes at the expense of long reorderings, which are often crucial to preserve the general meaning of a translated sentence. For instance, taking English as the target language, it is precisely the relative positioning of predicate arguments that determines their role, in the absence of case markers. Thus, a wrongly reordered verb with

minor impact on automatic scores, can be judged very badly by a human evaluator.

We will now describe two rule sets aimed at capturing these reordering phenomena.

### 4 Shallow syntax reordering rules

To compute the source reorderings, we use chunk-based rules following Bisazza and Federico (2010). Shallow syntax chunking is indeed a lighter and simpler task compared to full parsing, and it can be used to constrain the number of reorderings in a softer way. While rules based on full parses are generally deterministic, chunk-based rules are non-deterministic or fuzzy, as they generate several permutations for each matching sequence<sup>2</sup>. Besides defining a unique segmentation of the sentence, chunk annotation provides other useful information that can be used by the rules – namely chunk type and POS tags<sup>3</sup>.

For **Arabic-English** we apply the rules proposed by Bisazza and Federico (2010) aimed at transforming VS(O) sentences into SV(O). Reorderings are generated by moving each verb chunk (VC), alone or with its following chunk, by 1 to 6 chunks to the right. The maximum movement of each VC is limited to the position of the next VC, so that neighboring verb-reordering sequences may not overlap. This rule set was shown to cover most (99.5%) of the verb reorderings observed in a parallel news corpus, including those where the verb must be moved along with an adverbial or a complement.

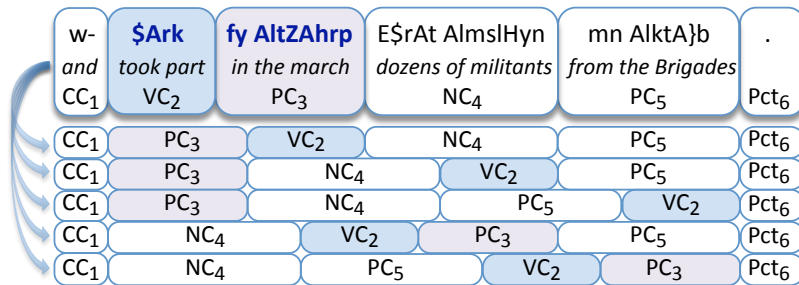
For **German-English** we propose a set of three rules<sup>4</sup> aimed at arranging the German constituents in SVO order:

- *infinitive*: move each infinitive VC right after a preceding punctuation;
- *subordinate*: if a VC is immediately followed by a punctuation, place it after a preceding subordinating conjunction (KOUS) or substitutive relative pronoun (PRELS);

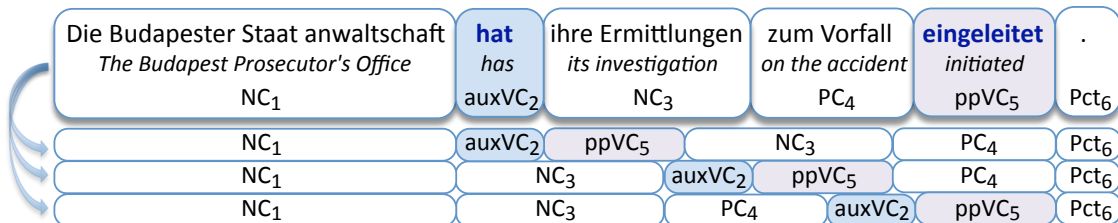
<sup>2</sup>Chunk annotation does not identify subject and complement boundaries, nor the relations among constituents that are needed to deterministically rearrange a sentence in SVO order.

<sup>3</sup>We use AMIRA (Diab et al., 2004) to annotate Arabic and Tree Tagger (Schmid, 1994) to annotate German.

<sup>4</sup>A similar rule set was previously used to produce chunk reordering lattices in (Hardmeier et al., 2010).



(a) Arabic VS(O) clause: five permutations



(b) German *broken* verb chunk: three permutations

Figure 1: Examples of chunk permutations generated by shallow syntax reordering rules. Chunk types: CC conjunction, VC verb (auxiliary/past participle), PC preposition, NC noun, Pct punctuation.

- *broken verb chunk*: join each finite VC (auxiliary or modal) with the nearest following non-finite VC (infinitive or participle). Place the resulting block in any position between the original position of the finite verb and that of the non-finite verb<sup>5</sup>.

The application of chunk reordering rules is illustrated by Fig. 1: in the Arabic sentence (a), the subject ‘dozens of militants’ is preceded by the main verb ‘took part’ and its argument ‘to the march’. The rules generate 5 permutations for one matching sequence (chunks 2 to 5), out of which the 5<sup>th</sup> is the best for translation. The German sentence (b) contains a broken VC with the inflected auxiliary ‘has’ separated from the past participle ‘initiated’. Here, the rules generate 3 permutations for the chunk sequence 2 to 5, corresponding to likely locations of the merged verb phrase, the 1<sup>st</sup> being optimal.

By construction, both rule sets generate a limited number of permutations per matching sequence: in

<sup>5</sup>To bound the number of reorderings, we use the following heuristics. In ‘*infinitive*’ at most 3 punctuations preceding the VC are considered. In ‘*subordinate*’ 1 to 3 chunks are left between the conjunction (or pronoun) and the moved VC to account for the subject. In ‘*broken VC*’ if the distance between the finite and non-finite verb is more than 10 chunks, only the first 5 and last 5 positions of the verb-to-verb span are considered.

Arabic at most 12 for each VC; in German at most 3 for each infinitive VC and for each VC-punctuation sequence, at most 10 for each broken VC. Empirically, this yields on average 22 reorderings per sentence in the NIST-MT Arabic benchmark dev06-NW and 3 on the WMT German benchmark test08. Arabic rules are indeed more noisy, which is not surprising as reordering is triggered by any verb chunk.

## 5 Reordering selection

The number of chunk-based reorderings per sentence varies according to the rule set, to the size of chunks and to the context. A high degree of fuzziness can complicate the decoding process, leaving too much work to the in-decoding reordering model. A solution to this problem is using an external model to score the rule-generated reorderings and discard the less probable ones. In such a way, a further part of reordering complexity is taken out of decoding.

At this end, instead of using a Support Vector Machine classifier as was done in (Bisazza et al., 2011), we apply *reordered n-gram models* that are lighter-weight and more suitable for a ranking task.

Differently from Feng et al. (2010), we train our models on partially reordered data and at the level of chunks. Chunks can be represented simply by their

type label (such as VC or NC), but also by a combination of the type and head word, to obtain finer lexicalized distributions. LMs trained on different chunk representations can also be applied jointly, by log-linear combination.

We perform reordering selection as follows:

1. Chunk-based reordering rules are applied deterministically to the *source* side of the parallel training data, using word alignment to choose the optimal permutation (“oracle reordering”)<sup>6</sup>.
2. One or several chunk-level 5-gram LMs are trained on such reordered data, using different chunk representation modes.
3. Reordering rules are applied to the test sentences and the resulting sets of rule-matching sequence permutations are scored by the LMs. The *n*-best reorderings of each rule-matching sequence are selected for translation.

In experiments not reported here, we obtained accurate rankings by scoring source permutations with a uniformly weighted combination of two LMs trained on chunk types and on chunk-type+headword, respectively. In particular, 3-best reorderings of each rule-matching sequence yield reordering recalls of 77.2% in Arabic and 89.3% in German.

## 6 Modified distortion matrices

We present here a novel technique to encode likely long reorderings of an input sentence, which can be seamlessly integrated into the PSMT framework.

During decoding, the distance between source positions is used for two main purposes: (i) generating a distortion penalty for the current hypothesis and (ii) determining the set of source positions that can be covered at the next hypothesis expansion. We can then tackle the coarseness of both distortion penalty and reordering constraints, by replacing the distance function with a function defined *ad hoc* for each input sentence.

Distortion can be thought of as a matrix assigning a positive integer to any ordered pair of source positions  $(s_x, s_y)$ . In the linear distortion model this is

<sup>6</sup>Following Bisazza and Federico (2010), the optimal reordering for a source sentence is the one that minimizes distortion in the word alignment to a target translation, measured by number of swaps and sum of distortion costs.

defined as:

$$D_L(s_x, s_y) = |s_y - s_x - 1|$$

so that moving to the right by 1 position costs 0 and by 2 positions costs 1. Moving to the left by 1 position costs 2 and by 2 positions costs 3, and so on. At the level of phrases, distortion is computed between the last word of the last translated phrase and the first word of the next phrase. We retain this equation as the core distortion function for our model. Then, we modify entries in the matrix such that the distortion cost is minimized for the decoding paths pre-computed with the reordering rules.

Given a source sentence and its set of rule-generated permutations, the linear distortion matrix is modified as follows:

1. non-monotonic jumps (i.e. ordered pairs  $(s_i, s_{i+1})$  such that  $s_{i+1} - s_i \neq 1$ ) are extracted from the permutations;
2. then, for each extracted pair, the corresponding point in the matrix is assigned the lowest possible distortion cost, that is 0 if  $s_i < s_{i+1}$  and 2 if  $s_i > s_{i+1}$ . We call these points *shortcuts*.

Although this technique is approximate and can overgenerate minimal-distortion decoding paths<sup>7</sup>, it practically works when the number of encoded permutations per sequence is limited. This makes modified distortion matrices particularly suitable to encode just those reorderings that are typically missed by phrase-based decoders (see Sect. 3).

Since in this work we use chunk-based rules, we also have to convert chunk-to-chunk jumps into word-to-word shortcuts. We propose two ways to do this, given an ordered pair of chunks  $(c_x, c_y)$ :

**mode  $\mathbf{A} \times \mathbf{A}$**  : create a shortcut from each word of  $c_x$  to each word of  $c_y$ ;

**mode  $\mathbf{L} \times \mathbf{F}$**  : create only one shortcut from the last word of  $c_x$  to the first of  $c_y$ .

The former solution admits more chunk-internal permutations with the same minimal distortion cost, whereas the latter implies that the first word of a re-ordered chunk is covered first and the last is covered last.

<sup>7</sup>In fact, any decoding path that includes a jump marked as shortcut benefits from the same distortion discount in that point.

orig: NC<sub>1</sub> auxVC<sub>2</sub> NC<sub>3</sub> PC<sub>4</sub> ppVC<sub>5</sub> Punc<sub>6</sub>  
 reo: NC<sub>1</sub> auxVC<sub>2</sub> ppVC<sub>5</sub> NC<sub>3</sub> PC<sub>4</sub> Punc<sub>6</sub>

	Die	Budapester	Staat	anwaltschaft	hat	ihre	Ermittlungen	zum	Vorfall	eingeleitet	.
<S>	0	1	2	3	4	5	6	7	8	9	10
Die		0	1	2	3	4	5	6	7	8	9
NC <sub>1</sub> Budapester	2		0	1	2	3	4	5	6	7	8
Staat	3	2		0	1	2	3	4	5	6	7
anwaltschaft	4	3	2		0	1	2	3	4	5	6
auxVC <sub>2</sub> hat	5	4	3	2		0	1	2	3	0	5
ihre	6	5	4	3	2		0	1	2	3	4
NC <sub>3</sub> Ermittlungen	7	6	5	4	3	2		0	1	2	3
zum	8	7	6	5	4	3	2		0	1	0
PC <sub>4</sub> Vorfall	9	8	7	6	5	4	3	2		0	0
ppVC <sub>5</sub> eingeleitet	10	9	8	7	6	2	2	3	2		0
Pct <sub>6</sub> .	11	10	9	8	7	6	5	4	3	2	

Figure 2: Modified distortion matrix (mode  $A \times A$ ) of the German sentence given in Fig. 1. The chunk reordering shown on top generates three *shortcuts* corresponding to the 0's and 2's highlighted in the matrix.

Fig. 2 shows the distortion matrix of the German sentence of Fig. 1, with starting positions as columns and landing positions as rows. Suppose we want to encode the reordering shown on top of Fig. 2, corresponding to the merging of the broken VC ‘hat ... eingeleitet’. This permutation contains three jumps: (2,5), (5,3) and (4,6). Converted to word-level in  $A \times A$  mode, these yield five word shortcuts<sup>8</sup>: one for the onward jump (2,5) assigned 0 distortion; two for the backward jump (5,3), assigned 2; and two for the onward jump (4,6), also assigned 0. The desired reordering is now attainable within a DL of 2 words instead of 5. The same process is then applied to other permutations of the sentence.

If compared to the word reordering lattices used by Bisazza and Federico (2010) and Andreas et al. (2011), modified distortion matrices provide a more compact, implicit way to encode likely reorderings in a sentence-specific fashion. Matrix representation does not require multiplication of nodes for the same

<sup>8</sup>In  $L \times F$  mode, instead, each chunk-to-chunk jump would yield exactly one word shortcut, for a total of three.

source word and is naturally compatible with the PSMT decoder’s standard reordering mechanisms.

## 7 Evaluation

In this section we evaluate the impact of modified distortion matrices on two news translation tasks.

Matrices were integrated into the Moses toolkit (Koehn et al., 2007) using a sentence-level XML markup. The list of word shortcuts for each sentence is provided as an XML tag that is parsed by the decoder to modify the distortion matrix just before starting the search. As usual, the distortion matrix is queried by the distortion penalty generator and by the hypothesis expander<sup>9</sup>.

### 7.1 Experimental setup

For **Arabic-English**, we use the union of all in-domain parallel corpora provided for the NIST-MT09 evaluation<sup>10</sup> for a total of 986K sentences, 31M English words. The target LM is trained on the English side of all available NIST-MT09 parallel data, UN included (147M words). For development and test, we use the newswire sections of the NIST benchmarks, hereby called dev06-NW, eval08-NW and eval09-NW: 1033, 813 and 586 sentences, respectively, each provided with four reference translations.

The **German-English** system is instead trained on WMT10 data: namely Europarl (v.5) plus News-commentary-2010 for a total of 1.6M parallel sentences, 43M English words. The target LM is trained on the monolingual news data provided for the constrained track (1133M words). For development and test, we use the WMT10 news benchmarks test08, test09 and test10: 2051, 2525 and 2489 sentences, respectively, with one reference translation.

Concerning pre-processing, we apply standard tokenization to the English data, while for Arabic we use our in-house tokenizer that removes diacritics and normalizes special characters. Arabic text is then segmented with AMIRA (Diab et al., 2004) according to the ATB scheme<sup>11</sup>. German tokenization

<sup>9</sup>Note that lexicalized reordering models use real word distances to compute the orientation class of a new hypothesis, thus they are not affected by changes in the matrix.

<sup>10</sup>That is everything except the small GALE corpus and the UN corpus. As reported by Green et al. (2010) the removal of UN data does not affect baseline performances on news test.

<sup>11</sup>The Arabic Treebank tokenization scheme isolates con-

and compound splitting is performed with Tree Tagger (Schmid, 1994) and the Gertwol morphological analyser (Koskenniemi and Haapalainen, 1994)<sup>12</sup>.

Using Moses we build competitive baselines on the training data described above. Word alignment is produced by the Berkeley Aligner (Liang et al., 2006). The decoder is based on the log-linear combination of a phrase translation model, a lexicalized reordering model, a 6-gram target language model, distortion cost, word and phrase penalties. The reordering model is a hierarchical phrase orientation model (Tillmann, 2004; Koehn et al., 2005; Galley and Manning, 2008) trained on all the available parallel data. We choose the hierarchical variant, as it was shown by its authors to outperform the default word-based on an Arabic-English task. Finally, for German, we enable the Moses option *monotone-at-punctuation* which forbids reordering across punctuation marks. The DL is initially set to 5 words for Arabic-English and to 10 for German-English. According to our experience, these are the optimal settings for the evaluated tasks. Feature weights are optimized by minimum error training (Och, 2003) on the development sets (dev06-NW and test08).

## 7.2 Translation quality and efficiency results

We evaluate translations with BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005). As these scores are only indirectly sensitive to word order, we also compute KRS or Kendall Reordering Score (Birch et al., 2010; Bisazza et al., 2011) which is a positive score based on the Kendall’s Tau distance between the source-output and source-reference permutations. To isolate the impact of our techniques on problematic reordering, we extract from each test set the sentences that got permuted by “oracle reordering” (see Sect. 5). These constitute about a half of the Arabic sentences, and about a third of the German. We refer to the KRS computed on these test *subsets* as KRS(R). Statistically significant differences are assessed by approximate randomization as in (Riezler and Maxwell, 2005)<sup>13</sup>.

Tab. 1 reports results obtained by varying the DL

junctions *w+* and *f+*, prepositions *l+*, *k+*, *b+*, future marker *s+*, pronominal suffixes, but not the article *A/+*.

<sup>12</sup><http://www2.lingsoft.fi/cgi-bin/gertwol>

<sup>13</sup>Translation scores and significance tests are computed with the tools *multeval* (Clark et al., 2011) and *sigf* (Padó, 2006).

and modifying the distortion function. To evaluate the reordering selection technique, we also compare the encoding of all rule-generated reorderings against only the 3 best per rule-matching sequence, as ranked by our best performing reordered LM (see end of Sect. 5). We mark the DL with a ‘+’ to denote that some longer jumps are being allowed by modified distortion. Run times refer to the translation of the first 100 sentences of eval08-NW and test09 by a 4-core processor.

**Arabic-English.** As anticipated, raising the DL does not improve, but rather worsen performances. The decrease in BLEU and METEOR reported with DL=8 is not significant, but the decrease in KRS is both significant and large. Efficiency is heavily affected, with a 42% increase of the run time.

Results in the row “allReo” are obtained by encoding all the rule-generated reorderings in  $L \times F$  chunk-to-word conversion mode. Except for some gains in KRS reported on eval08-NW, most of the scores are lower or equal to the baseline. Such inconsistent behaviour is probably due to the low precision of the Arabic rule set, pointed out in Sect. 4.

Finally, we arrive to the performance of 3-best reorderings per sequence. With  $L \times F$  we obtain several improvements, but it’s with  $A \times A$  that we are able to beat the baseline according to all metrics. BLEU and METEOR improvements are rather small but significant and consistent across test sets, the best gain being reported on eval09-NW (+.9 BLEU). Most importantly, substantial word order improvements are achieved on both full test sets (+.7/+6 KRS) and selected subsets (+.7/+6 KRS(R)). According to these figures, word order is affected only in the sentences that contain problematic reordering. This is good evidence, suggesting that the decoder does not get “confused” by spurious shortcuts.

Looking at run times, we can say that modified distortion matrices are a very efficient way to address long reordering. Even when all the generated reorderings are encoded, translation time increases only by 4%. Reordering selection naturally helps to further reduce decoding overload. As for conversion modes,  $A \times A$  yields slightly higher run times than  $L \times F$  because it generates more shortcuts.

**German-English.** In this task we manage to improve translation quality with a setting that is almost



(a) Arabic to English

Distortion Function	DL	eval08-nw				eval09-nw				runtime (s)
		bleu	met	krs	krs(R)	bleu	met	krs	krs(R)	
† plain [ <i>baseline</i> ]	5	44.5	34.9	81.6	82.9	49.9	38.0	84.1	84.4	1038
plain	8	44.2 <sup>°</sup>	34.8	80.7 <sup>•</sup>	82.2 <sup>•</sup>	49.8	37.9	83.3 <sup>•</sup>	83.5 <sup>•</sup>	1470
† modified: allReo, L×F	5+	44.4	34.9	82.2 <sup>•</sup>	83.7 <sup>•</sup>	49.9	37.8 <sup>•</sup>	84.3	84.4	1078
modified: 3bestReo, L×F	5+	44.5	35.1 <sup>•</sup>	82.3 <sup>•</sup>	83.5 <sup>•</sup>	50.7 <sup>•</sup>	38.1	84.8 <sup>•</sup>	85.0 <sup>•</sup>	1052
† modified: 3bestReo, A×A	5+	44.8 <sup>°</sup>	35.1 <sup>•</sup>	82.3 <sup>•</sup>	83.6 <sup>•</sup>	50.8 <sup>•</sup>	38.2 <sup>•</sup>	84.7 <sup>•</sup>	85.0 <sup>•</sup>	1072

(b) German to English

Distortion Function	DL	test09				test10				runtime (s)
		bleu	met	krs	krs(R)	bleu	met	krs	krs(R)	
† plain [ <i>baseline</i> ]	10	18.8	27.5	65.8	66.7	20.1	29.4	68.7	68.9	629
plain	20	18.4 <sup>•</sup>	27.4 <sup>•</sup>	63.6 <sup>•</sup>	65.2 <sup>•</sup>	19.8 <sup>•</sup>	29.3 <sup>•</sup>	66.3 <sup>•</sup>	66.6 <sup>•</sup>	792
plain	4	18.4 <sup>•</sup>	27.4 <sup>•</sup>	67.3 <sup>•</sup>	66.9	19.6 <sup>•</sup>	29.1 <sup>•</sup>	70.2 <sup>•</sup>	69.6 <sup>•</sup>	345
† modified: allReo, L×F	4+	19.1 <sup>•</sup>	27.6 <sup>•</sup>	67.6 <sup>•</sup>	68.1 <sup>•</sup>	20.4 <sup>•</sup>	29.4	70.6 <sup>•</sup>	70.7 <sup>•</sup>	352
modified: 3bestReo, L×F	4+	19.2 <sup>•</sup>	27.7 <sup>•</sup>	67.4 <sup>•</sup>	68.1 <sup>•</sup>	20.4 <sup>•</sup>	29.4	70.4 <sup>•</sup>	70.6 <sup>•</sup>	351
† modified: 3bestReo, A×A	4+	19.2 <sup>•</sup>	27.7 <sup>•</sup>	67.4 <sup>•</sup>	68.4 <sup>•</sup>	20.6 <sup>•</sup>	29.5 <sup>°</sup>	70.4 <sup>•</sup>	70.7 <sup>•</sup>	357

Table 1: Impact of modified distortion matrices on translation quality, measured with BLEU, METEOR and KRS (all in percentage form, higher scores mean higher quality). The settings used for weight tuning are marked with †. Statistically significant differences wrt the baseline are marked with • at the  $p \leq .05$  level and ° at the  $p \leq .10$  level.

twice as fast as the baseline. As shown by the first part of the table, the best baseline results are obtained with a rather high DL, that is 10 (only KRS improves with a lower DL). However, with modified distortion, the best results according to all metrics are obtained with a DL of 4.

Looking at the rest of the table, we see that reordering selection is not as crucial as in Arabic-English. This is in line with the properties of the more precise German reordering rule set (two rules out of three generate at most 3 reorderings per sequence). Considering all scores, the last setting (3-best reordering and A×A) appears as the best one, achieving the following gains over the baseline: +.4/+1.5 BLEU, +.2/+1.1 METEOR, +1.6/+1.7 KRS and +1.7/+1.8 KRS(R). The agreement observed among such diverse metrics makes us confident about the goodness of the approach.

## 8 Conclusions

In Arabic-English and German-English, long reordering concentrates on specific patterns describable by a small number of linguistic rules. By means of non-deterministic chunk reordering rules, we have generated likely permutations of the test

sentences and ranked them with n-gram LMs trained on pre-reordered data. We have then introduced the notion of modified distortion matrices to naturally encode a set of likely reorderings in the decoder input. Modified distortion allows for a finer and more linguistically informed definition of the search space, which is reflected in better translation outputs and more efficient decoding.

We expect that further improvements may be achieved by refining the Arabic reordering rules with specific POS tags and lexical cues. We also plan to evaluate modified distortion matrices in conjunction with a different type of in-decoding reordering model such as the one proposed by Green et al. (2010). Finally, we may try to exploit not only the ranking, but also the scores produced by the re-ordered LMs, as an additional decoding feature.

## Acknowledgments

This work was supported by the T4ME network of excellence (IST-249119) funded by the European Commission DG INFSO through the 7<sup>th</sup> Framework Programme. We thank Christian Hardmeier for helping us define the German reordering rules, and the anonymous reviewers for valuable suggestions.

## References

- Yaser Al-Onaizan and Kishore Papineni. 2006. Distortion models for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 529–536, Sydney, Australia, July. Association for Computational Linguistics.
- Jacob Andreas, Nizar Habash, and Owen Rambow. 2011. Fuzzy syntactic reordering for phrase-based statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 227–236, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Alexandra Birch, Miles Osborne, and Phil Blunsom. 2010. Metrics for MT evaluation: evaluating reordering. *Machine Translation*, 24(1):15–26.
- Arianna Bisazza and Marcello Federico. 2010. Chunk-based verb reordering in VSO sentences for Arabic-English statistical machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR*, pages 241–249, Uppsala, Sweden, July. Association for Computational Linguistics.
- Arianna Bisazza, Daniele Pighin, and Marcello Federico. 2011. Chunk-lattices for verb reordering in Arabic-English statistical machine translation. *Machine Translation*, Published Online.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Jonathan Clark, Chris Dyer, Alon Lavie, and Noah Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proceedings of the Association for Computational Linguistics*, ACL 2011, Portland, Oregon, USA. Association for Computational Linguistics. accepted; available at <http://www.cs.cmu.edu/~jhclark/pubs/significance.pdf>.
- Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 531–540, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Josep M. Crego and Nizar Habash. 2008. Using shallow syntax information to improve word alignment and reordering for smt. In *StatMT '08: Proceedings of the Third Workshop on Statistical Machine Translation*, pages 53–61, Morristown, NJ, USA. Association for Computational Linguistics.
- Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 149–152, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.
- Jakob Elming and Nizar Habash. 2009. Syntactic reordering for English-Arabic phrase-based machine translation. In *Proceedings of the EACL 2009 Workshop on Computational Approaches to Semitic Languages*, pages 69–77, Athens, Greece, March. Association for Computational Linguistics.
- Minwei Feng, Arne Mauser, and Hermann Ney. 2010. A source-side decoding sequence model for statistical machine translation. In *Conference of the Association for Machine Translation in the Americas (AMTA)*, Denver, Colorado, USA.
- Michel Galley and Christopher D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Morristown, NJ, USA. Association for Computational Linguistics.
- Spence Green, Michel Galley, and Christopher D. Manning. 2010. Improved models of distortion cost for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 867–875, Los Angeles, California. Association for Computational Linguistics.
- Nizar Habash. 2007. Syntactic preprocessing for statistical machine translation. In Bente Maegaard, editor, *Proceedings of the Machine Translation Summit XI*, pages 215–222, Copenhagen, Denmark.
- Christian Hardmeier, Arianna Bisazza, and Marcello Federico. 2010. FBK at WMT 2010: Word lattices for morphological reduction and chunk-based reordering. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR*, pages 88–92, Uppsala, Sweden, July. Association for Computational Linguistics.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceed-*

- ings of *HLT-NAACL 2003*, pages 127–133, Edmonton, Canada.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh system description for the 2005 IWSLT speech translation evaluation. In *Proc. of the International Workshop on Spoken Language Translation*, October.
- 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 *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic.
- Kimmo Koskenniemi and Mariikka Haapalainen, 1994. *GERTWOL – Lingsoft Oy*, chapter 11, pages 121–140. Roland Hausser, Niemeyer, Tübingen.
- Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 104–111, New York City, USA, June. Association for Computational Linguistics.
- Jan Niehues and Muntsin Kolss. 2009. A POS-based model for long-range reorderings in SMT. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 206–214, Athens, Greece, March. Association for Computational Linguistics.
- Franz Josef Och. 2002. *Statistical Machine Translation: From Single-Word Models to Alignment Templates*. Ph.D. thesis, RWTH Aachen, Germany.
- Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In Erhard Hinrichs and Dan Roth, editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.
- Sebastian Padó, 2006. *User’s guide to sigf: Significance testing by approximate randomisation*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 311–318, Philadelphia, PA.
- Stefan Riezler and John T. Maxwell. 2005. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 57–64, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*.
- Christoph Tillmann. 2004. A Unigram Orientation Model for Statistical Machine Translation. In *Proceedings of the Joint Conference on Human Language Technologies and the Annual Meeting of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL)*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.
- Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of Coling 2004*, pages 508–514, Geneva, Switzerland, Aug 23–Aug 27. COLING.
- Sirvan Yahyaei and Christof Monz. 2010. Dynamic distortion in a discriminative reordering model for statistical machine translation. In *International Workshop on Spoken Language Translation (IWSLT)*, Paris, France.
- Kenji Yamada. 2002. *A syntax-based translation model*. Ph.D. thesis, Department of Computer Science, University of Southern California, Los Angeles.
- Richard Zens and Hermann Ney. 2006. Discriminative reordering models for statistical machine translation. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 55–63, New York City, June. Association for Computational Linguistics.
- R. Zens, F. J. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In *25th German Conference on Artificial Intelligence (KI2002)*, pages 18–32, Aachen, Germany. Springer Verlag.
- Yuqi Zhang, Richard Zens, and Hermann Ney. 2007. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 1–8, Rochester, New York, April. Association for Computational Linguistics.
- Andreas Zollmann, Ashish Venugopal, Franz Och, and Jay Ponte. 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 1145–1152, Manchester, UK, August. Coling 2008 Organizing Committee.

# Semantic Parsing with Bayesian Tree Transducers

Bevan Keeley Jones\*†

b.k.jones@sms.ed.ac.uk

Mark Johnson†

Mark.Johnson@mq.edu.au

Sharon Goldwater\*

sgwater@inf.ed.ac.uk

\* School of Informatics  
University of Edinburgh  
Edinburgh, EH8 9AB, UK

† Department of Computing  
Macquarie University  
Sydney, NSW 2109, Australia

## Abstract

Many semantic parsing models use tree transformations to map between natural language and meaning representation. However, while tree transformations are central to several state-of-the-art approaches, little use has been made of the rich literature on tree automata. This paper makes the connection concrete with a tree transducer based semantic parsing model and suggests that other models can be interpreted in a similar framework, increasing the generality of their contributions. In particular, this paper further introduces a variational Bayesian inference algorithm that is applicable to a wide class of tree transducers, producing state-of-the-art semantic parsing results while remaining applicable to any domain employing probabilistic tree transducers.

## 1 Introduction

Semantic parsing is the task of mapping natural language sentences to a formal representation of meaning. Typically, a system is trained on pairs of natural language sentences (NLs) and their meaning representation expressions (MRs), as in figure 1(a), and the system must generalize to novel sentences.

Most semantic parsing models rely on an assumption of structural similarity between MR and NL. Since strict isomorphism is overly restrictive, this assumption is often relaxed by applying transformations. Several approaches assume a tree structure to the NL, MR, or both (Ge and Mooney, 2005; Kate and Mooney, 2006; Wong and Mooney, 2006; Lu et al., 2008; Börschinger et al., 2011), and often in-

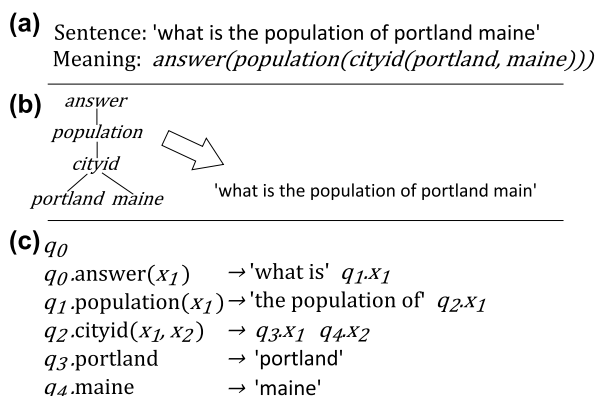


Figure 1: (a) An example sentence/meaning pair, (b) a tree transformation based mapping, and (c) a tree transducer that performs the mapping.

volve tree transformations either between two trees or a tree and a string.

The tree transducer, a formalism from automata theory which has seen interest in machine translation (Yamada and Knight, 2001; Graehl et al., 2008) and has potential applications in many other areas, is well suited to formalizing such tree transformation based models. Yet, while many semantic parsing systems resemble the formalism, each was proposed as an independent model requiring custom algorithms, leaving it unclear how developments in one line of inquiry relate to others. We argue for a unifying theory of tree transformation based semantic parsing by presenting a tree transducer model and drawing connections to other similar systems.

We make a further contribution by bringing to tree transducers the benefits of the Bayesian framework for principled handling of data sparsity and

prior knowledge. Graehl et al. (2008) present an EM training procedure for top down tree transducers, but while there are Bayesian approaches to string transducers (Chiang et al., 2010) and PCFGs (Kurihara and Sato, 2006), there has yet to be a proposal for Bayesian inference in *tree* transducers. Our variational algorithm produces better semantic parses than EM while remaining general to a broad class of transducers appropriate for other domains.

In short, our contributions are three-fold: we present a new state-of-the-art semantic parsing model, propose a broader theory for tree transformation based semantic parsing, and present a general inference algorithm for the tree transducer framework. We recommend the last of these as just one benefit of working within a general theory: contributions are more broadly applicable.

## 2 Meaning representations and regular tree grammars

In semantic parsing, an MR is typically an expression from a machine interpretable language (e.g., a database query language or a logical language like Prolog). In this paper we assume MRs can be represented as trees, either by pre-parsing or because they are already trees (often the case for functional languages like LISP).<sup>1</sup> More specifically, we assume the MR language is a regular tree language.

A regular tree grammar (RTG) closely resembles a context free grammar (CFG), and is a way of describing a language of trees. Formally, define  $T_\Sigma$  as the set of trees with symbols from alphabet  $\Sigma$ , and  $T_\Sigma(\mathcal{A})$  as the set of all trees in  $T_{\Sigma \cup \mathcal{A}}$  where symbols from  $\mathcal{A}$  only occur at the leaves. Then an RTG is a tuple  $(\mathcal{Q}, \Sigma, q_{start}, \mathcal{R})$ , where  $\mathcal{Q}$  is a set of states,  $\Sigma$  is an alphabet,  $q_{start} \in \mathcal{Q}$  is the initial state, and  $\mathcal{R}$  is a set of grammar rules of the form  $q \rightarrow t$ , where  $q$  is a state from  $\mathcal{Q}$  and  $t$  is a tree from  $T_\Sigma(\mathcal{Q})$ .

A rule typically consists of a parent state (left) and its child states and output symbol (right). We indicate states using all capital letters:

NUM  $\rightarrow$  population(PLACE).

Intuitively, an RTG is a CFG where the yield of every parse is itself a tree. In fact, for any CFG  $G$ , it

<sup>1</sup>See Liang et al. (2011) for work in representing lambda calculus expressions with trees.

is straightforward to produce a corresponding RTG that generates the set of parses of  $G$ . Consequently, while we assume we have an RTG for the MR language, there is no loss of generality if the MR language is actually context free.

## 3 Weighted root-to-frontier, linear, non-deleting tree-to-string transducers

Tree transducers (Rounds, 1970; Thatcher, 1970) are generalizations of finite state machines that operate on trees. Mirroring the branching nature of its input, the transducer may simultaneously transition to several successor states, assigning a separate state to each subtree.

There are many classes of transducer with different formal properties (Knight and Graehl, 2005; Maletti et al., 2009). Figure 1(c) is an example of a root-to-frontier, linear, non-deleting tree-to-string transducer. It is defined using rules where the left hand side identifies a state of the transducer and a fragment of the input tree, and the right hand side describes a portion of the output string. Variables  $x_i$  stand for entire sub-trees, and state-variable pairs  $q_j.x_i$  stand for strings produced by applying the transducer starting at state  $q_j$  to subtree  $x_i$ . Figure 1(b) illustrates an application of the transducer, taking the tree on the left as input and outputting the string on the right.

Formally, a weighted root-to-frontier, tree-to-string transducer is a 5-tuple  $(\mathcal{Q}, \Sigma, \Delta, q_{start}, \mathcal{R})$ .  $\mathcal{Q}$  is a finite set of states,  $\Sigma$  and  $\Delta$  are the input and output alphabets,  $q_{start}$  is the start state, and  $\mathcal{R}$  is the set of rules. Denote a pair of symbols,  $a$  and  $b$  by  $a.b$ , the cross product of two sets  $\mathcal{A}$  and  $\mathcal{B}$  by  $\mathcal{A}.\mathcal{B}$ , and let  $\mathcal{X}$  be the set of variables  $\{x_0, x_1, \dots\}$ . Then, each rule  $r \in \mathcal{R}$  is of the form  $[q.t \rightarrow u].v$ , where  $v \in \mathbb{R}^{\geq 0}$  is the rule weight,  $q \in \mathcal{Q}$ ,  $t \in T_\Sigma(\mathcal{X})$ , and  $u$  is a string in  $(\Delta \cup \mathcal{Q}.\mathcal{X})^*$  such that every  $x \in \mathcal{X}$  in  $u$  also occurs in  $t$ .

We say  $q.t$  is the left hand side of rule  $r$  and  $u$  its right hand side. The transducer is *linear* iff no variable appears more than once on the right hand side. It is *non-deleting* iff all variables on the left hand side also occur on the right hand side. In this paper we assume that every tree  $t$  on the left hand side is either a single variable  $x_0$  or of the form  $\sigma(x_0, \dots x_n)$ , where  $\sigma \in \Sigma$  (i.e., it is a tree of depth  $\leq 1$ ).

A weighted tree transducer may define a probability distribution, either a joint distribution over input and output pairs or a conditional distribution of the output given the input. Here, we will use joint distributions, which can be defined by ensuring that the weights of all rules with the same state on the left-hand side sum to one. In this case, it can be helpful to view the transducer as simultaneously generating both the input and output, rather than the usual view of mapping input trees into output strings. A joint distribution allows us to model with a single machine both the input and output languages, which is important during decoding when we want to infer the input given the output.

#### 4 A generative model of semantic parsing

Like the hybrid tree semantic parser (Lu et al., 2008) and the synchronous grammar based WASP (Wong and Mooney, 2006), our model simultaneously generates the input MR tree and the output NL string. The MR tree is built up according to the provided MR grammar, one grammar rule at a time. Coupled with the application of the MR rule, similar CFG-like productions are applied to the NL side, repeated until both the MR and NL are fully generated. In each step, we select an MR rule and then build the NL by first choosing a pattern with which to expand it and then filling out that pattern with words drawn from a unigram distribution.

This kind of coupled generative process can be naturally formalized with tree transducer rules, where the input tree fragment on the left side of each rule describes the derivation of the MR and the right describes the corresponding NL derivation.

For a simple example of a tree-to-string transducer rule consider

$$q.\text{population}(x_1) \rightarrow \text{'population of'} q.x_1 \quad (1)$$

which simultaneously generates tree fragment  $\text{population}(x_1)$  on the left and sub-string “population of  $q.x_1$ ” on the right. Variable  $x_1$  stands for an MR subtree under  $\text{population}$ , and, on the right, state-variable pair  $q.x_1$  stands for the NL substring generated while processing subtree  $x_1$  starting from  $q$ . While this rule can serve as a single step of an MR-to-NL map such as the example transducer shown in Figure 1(c), such rules do not model the

$$\text{NUM} \rightarrow \text{population}(\text{PLACE}) \quad (m)$$

$$\text{PLACE} \rightarrow \text{cityid}(\text{CITY}, \text{STATE}) \quad (r)$$

$$\text{CITY} \rightarrow \text{portland} \quad (u)$$

$$\text{STATE} \rightarrow \text{maine} \quad (v)$$

$$q_{m,1}^{\text{MR}}.x_1 \rightarrow q_r^{\text{NL}}.x_1 \quad (2)$$

$$q_{r,1}^{\text{MR}}.x_1 \rightarrow q_u^{\text{NL}}.x_1$$

$$q_{r,2}^{\text{MR}}.x_1 \rightarrow q_v^{\text{NL}}.x_1$$

$$q_m^{\text{NL}}.\text{population}(w_1, x_1, w_2) \rightarrow q_m^{\text{W}}.w_1 q_{m,1}^{\text{MR}}.x_1 q^{\text{END}}.w_2 \quad (3)$$

$$q_r^{\text{NL}}.\text{cityid}(w_1, x_1, w_2, x_2, w_3) \rightarrow q^{\text{END}}.w_1 q_{r,2}^{\text{MR}}.x_2 q_r^{\text{W}}.w_2 q_{r,1}^{\text{MR}}.x_1 q^{\text{END}}.w_3 \quad (4)$$

$$q_m^{\text{W}}.w_1 \rightarrow \text{'population'} q_m^{\text{W}}.w_1 \quad (5)$$

$$q_m^{\text{W}}.w_1 \rightarrow \text{'of'} q_m^{\text{W}}.w_1$$

$$q_m^{\text{W}}.w_1 \rightarrow \dots q_m^{\text{W}}.w_1$$

$$q_m^{\text{W}}.w_1 \rightarrow \text{'of'} q^{\text{END}}.w_1 \quad (6)$$

$$q_m^{\text{W}}.w_1 \rightarrow \dots q^{\text{END}}.w_1$$

$$q^{\text{END}}.W \rightarrow \epsilon \quad (7)$$

Figure 2: Examples of transducer rules (bottom) that generate MR and NL associated with MR rules  $m-v$  (top). Transducer rule 2 selects MR rule  $r$  from the MR grammar. Rule 3 simultaneously writes the MR associated with rule  $m$  and chooses an NL pattern (as does 4 for  $r$ ). Rules 5-7 generate the words associated with  $m$  according to a unigram distribution specific to  $m$ .

grammaticality of the MR and lack flexibility since sub-strings corresponding to a given tree fragment must be completely pre-specified. Instead, we break transductions down into a three stage process of choosing the (i) MR grammar rule, (ii) NL expansion pattern, and (iii) individual words according to a unigram distribution. Such a decomposition incorporates independence assumptions that improve generalizability. See Figure 2 for example rules from our transducer and Figure 3 for a derivation.

To ensure that only grammatical MRs are generated, each state of our transducer encodes the identity of exactly one MR grammar rule. Transitions between  $q^{\text{MR}}$  and  $q^{\text{NL}}$  states implicitly select the embedded rule. For instance, rule 2 in Figure 2 selects

MR grammar rule  $r$  to expand the  $i^{th}$  child of the parent produced by rule  $m$ . Aside from ensuring the *grammaticality* of the generated MR, rules of this type also model the *probability* of the MR, conditioning the probability of a rule both on the parent rule and the index of the child being expanded. Thus, parent state  $q_{m,1}^{MR}$  encodes not only the identity of rule  $m$ , but also the child index, 1 in this case.

Once the MR rule is selected,  $q^{NL}$  states are applied to select among rules such as 3 and 4 to generate the MR entity and choose the NL expansion pattern. These rules determine the word order of the language by deciding (i) whether or not to generate words in a given location and (ii) where to insert the result of processing each MR subtree. Decision (i) is made by either transitioning to state  $q_r^W$  to generate words or to  $q^{END}$  to generate the empty string. Decision (ii) is made with the order of  $x_i$ 's on the right hand side. Rule 4 illustrates the case where *portland* and *maine* in *cityid(portland, maine)* would be realized in reverse order as "maine ... portland".

The particular set of patterns that appear on the right of rules such as 3 embodies the binary word attachment decisions and the particular permutation of  $x_i$  in the NL. We allow words to be generated at the beginning and end of each pattern and between the  $x_i$ s. Thus, rule 4 is just one of 16 such possible patterns (3 binary decisions and 2 permutations), while rule 3 is one of 4. We instantiate all such rules and allow the system to learn weights for them according to the language of the training data.

Finally, the NL is filled out with words chosen according to a unigram distribution, implemented in a PCFG-like fashion, using a different rule for each word which recursively chooses the next word until a string termination rule is reached.<sup>2</sup> Generating word sequence "population of" entails first choosing rule 5 in Figure 2. State  $q_r^W$  is then recursively applied to choose rule 6, generating "of" at the same time as deciding to terminate the string by transitioning to a new state  $q^{END}$  which deterministically concludes by writing the empty string  $\epsilon$ .

On the MR side, rules 5-7 do very little: the tree on the left side of rules 5 and 6 consists entirely of a

<sup>2</sup>There are roughly 25,000 rules in the transducers in our experiments, and the majority of these implement the unigram word distributions since every entity in the MR may potentially produce any of the words it is paired with in training.

subtree variable  $w_1$ , indicating that nothing is generated in the MR. Rule 7 subsequently generates these subtrees as  $W$  symbols, marking corresponding locations where words might be produced in the NL, which are later removed during post processing.<sup>3</sup>

Figure 3(b) illustrates the coupled generative process. At each step of the derivation, an MR rule is chosen to expand a node of the MR tree, and then a corresponding part of the NL is expanded. Step 1.1 of the example chooses MR rule  $m$ ,  $NUM \rightarrow population(PLACE)$ . Transducer rule 3 then generates *population* in the MR (shown in the left column) at the same time as choosing an NL expansion pattern (Step 1.2) which is subsequently filled out with specific words "population" (1.3) and "of" (1.4).

This coupled derivation can be represented by a tree, shown in Figure 3(c), which explicitly represents the dependency structure of the coupled MR and NL (a simplified version is shown in (d) for clarity). In our transducer, which defines a joint distribution over both the MR and NL, the probability of a rule is conditioned on the parent state. Since each state encodes an MR rule, MR rule specific distributions are learned for both the words and their order.

## 5 Relation to existing models

The tree transducer model can be viewed either as a generative procedure for building up two separate structures or as a transformative machine that takes one as input and produces another as output. Different semantic parsing approaches have taken one or the other view, and both can be captured in this single framework.

WASP (Wong and Mooney, 2006) is an example of the former perspective, coupling the generation of the MR and NL with a synchronous grammar, a formalism closely related to tree transducers. The most significant difference from our approach is that they use machine translation techniques for automatically extracting rules from parallel corpora; similar techniques can be applied to tree transducers (Galley et al., 2004). In fact, synchronous grammars and tree transducers can be seen as instances of the same more general class of automata (Shieber,

<sup>3</sup>The addition of  $W$  symbols is a convenience; it is easier to design transducer rules where every substring on the right side corresponds to a subtree on the left.

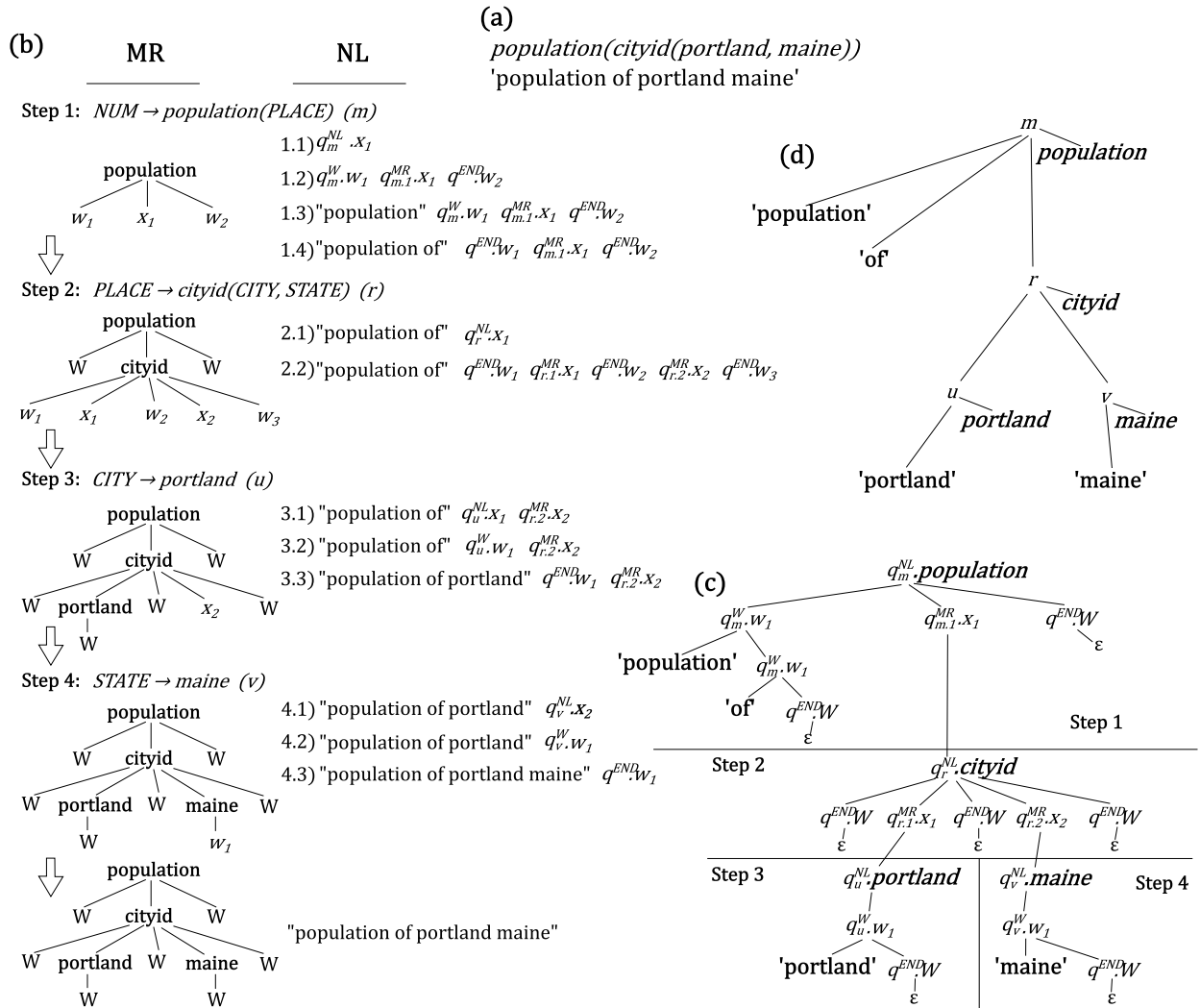


Figure 3: Coupled derivation of an (MR, NL) pair. At each step an MR grammar rule is chosen to expand the MR and the corresponding portion of the NL is then generated. Symbols  $W$  stand for locations in the tree corresponding to substrings of the output and are removed in a post-processing step. (a) The (MR, NL) pair. (b) Step by step derivation. (c) The same derivation shown in tree form. (d) The underlying dependency structure of the derivation.

2004). Rather than argue for one or the other, we suggest that other approaches could also be interpreted in terms of general model classes, grounding them in a broader base of theory.

The hybrid tree model (Lu et al., 2008) takes a transformative perspective that is in some ways more similar to our model. In fact, there is a one-to-one relationship between the multinomial parameters of the two models. However, they represent the MR and NL with a single tree and apply tree walking algorithms to extract them. Furthermore, they implement a custom training procedure for searching over the potential MR transformations. The tree

transducer, on the other hand, naturally captures the same probabilistic dependencies while maintaining the separation between MR and NL, and further allows us to build upon a larger body of theory.

KRISP (Kate and Mooney, 2006) uses string classifiers to label substrings of the NL with entities from the MR. To focus search, they impose an ordering constraint based on the structure of the MR tree, which they relax by allowing the re-ordering of sibling nodes and devise a procedure for recovering the MR from the permuted tree. This procedure corresponds to backward-application in tree transducers, identifying the most likely input tree given a



particular output string.

SCISSOR (Ge and Mooney, 2005) takes syntactic parses rather than NL strings and attempts to translate them into MR expressions. While few semantic parsers attempt to exploit syntactic information, there are techniques from machine translation for using tree transducers to map between parsed parallel corpora, and these techniques could likely be applied to semantic parsing.

Börschinger et al. (2011) argue for the PCFG as an alternative model class, permitting conventional grammar induction techniques, and tree transducers are similar enough that many techniques are applicable to both. However, the PCFG is less amenable to conceptualizing correspondences between parallel structures, and their model is more restrictive, only applicable to domains with finite MR languages, since their non-terminals encode entire MRs. The tree transducer framework, on the other hand, allows us to condition on individual MR rules.

## 6 Variational Bayes for tree transducers

As seen in the example in Figure 3(c), tree transducers not only operate on trees, their derivations are themselves trees, making them amenable to dynamic programming and an EM training procedure resembling inside-outside (Graehl et al., 2008). EM assigns zero probability to events not seen in the training data, however, limiting the ability to generalize to novel items. The Bayesian framework offers an elegant solution to this problem, introducing a prior over rule weights which simultaneously ensures that all rules receive non-zero probability and allows the incorporation of prior knowledge and intuitions. Unfortunately, the introduction of a prior makes exact inference intractable, so we use an approximate method, variational Bayesian inference (Bishop, 2006), deriving an algorithm similar to that for PCFGs (Kurihara and Sato, 2006).

The tree transducer defines a joint distribution over the input  $y$ , output  $w$ , and their derivation  $x$  as the product of the weights of the rules appearing in  $x$ . That is,

$$p(y, x, w|\theta) = \prod_{r \in \mathcal{R}} \theta(r)^{c_r(x)}$$

where  $\theta$  is the set of multinomial parameters,  $r$  is a transducer rule,  $\theta(r)$  is its weight, and  $c_r(x)$  is the

number of times  $r$  appears in  $x$ . In EM, we are interested in the point estimate for  $\theta$  that maximizes  $p(\mathcal{Y}, \mathcal{W}|\theta)$ , where  $\mathcal{Y}$  and  $\mathcal{W}$  are the  $N$  input-output pairs in the training data. In the Bayesian setting, however, we place a symmetric Dirichlet prior over  $\theta$  and estimate a posterior distribution over both  $\mathcal{X}$  and  $\theta$ .

$$\begin{aligned} p(\theta, \mathcal{X}|\mathcal{Y}, \mathcal{W}) &= \frac{p(\mathcal{Y}, \mathcal{X}, \mathcal{W}, \theta)}{p(\mathcal{Y}, \mathcal{W})} \\ &= \frac{p(\theta) \prod_{i=1}^N p(y_i, x_i, w_i|\theta)}{\int p(\theta) \prod_{i=1}^N \sum_{x \in \mathcal{X}_i} p(y_i, x, w_i|\theta) d\theta} \end{aligned}$$

Since the integral in the denominator is intractable, we look for an appropriate approximation  $q(\theta, \mathcal{X}) \approx p(\theta, \mathcal{X}|\mathcal{Y}, \mathcal{W})$ . In particular, we assume the rule weights and the derivations are independent, i.e.,  $q(\theta, \mathcal{X}) = q(\theta)q(\mathcal{X})$ . The basic idea is then to define a lower bound  $\mathcal{F} \leq \ln p(\mathcal{Y}, \mathcal{W})$  in terms of  $q$  and then apply the calculus of variations to find a  $q$  that maximizes  $\mathcal{F}$ .

$$\begin{aligned} \ln p(\mathcal{Y}, \mathcal{W}|\alpha) &= \ln E_q \left[ \frac{p(\mathcal{Y}, \mathcal{X}, \mathcal{W}|\theta)}{q(\theta, \mathcal{X})} \right] \\ &\geq E_q \left[ \ln \frac{p(\mathcal{Y}, \mathcal{X}, \mathcal{W}|\theta)}{q(\theta, \mathcal{X})} \right] = \mathcal{F}, \end{aligned}$$

Applying our independence assumption, we arrive at the following expression for  $\mathcal{F}$ , where  $\theta_t$  is the particular parameter vector corresponding to the rules with parent state  $t$ :

$$\begin{aligned} \mathcal{F} &= \sum_{t \in \mathcal{Q}} (E_{q(\theta_t)}[\ln p(\theta_t|\alpha_t)] - E_{q(\theta_t)}[\ln q(\theta_t)]) \\ &+ \sum_{i=1}^N (E_q[\ln p(w_i, x_i, y_i|\theta)] - E_{q(x_i)}[\ln q(x_i)]). \end{aligned}$$

We find the  $q(\theta_t)$  and  $q(x_i)$  that maximize  $\mathcal{F}$  by taking derivatives of the Lagrangian, setting them to zero, and solving, which yields:

$$\begin{aligned} q(\theta_t) &= \text{Dirichlet}(\theta_t|\hat{\alpha}_t) \\ q(x_i) &= \frac{\prod_{r \in \mathcal{R}} \hat{\theta}(r)^{c_r(x_i)}}{\sum_{x \in \mathcal{X}_i} \prod_{r \in \mathcal{R}} \hat{\theta}(r)^{c_r(x)}} \end{aligned}$$

where

$$\begin{aligned} \hat{\alpha}(r) &= \alpha(r) + \sum_i E_{q(x_i)}[c_r(x_i)] \\ \hat{\theta}(r) &= \exp \left( \Psi(\hat{\alpha}(r)) - \Psi \left( \sum_{r:s(r)=t} \hat{\alpha}(r) \right) \right). \end{aligned}$$

The parameters of  $q(\theta_t)$  are defined with respect to  $q(x_i)$  and the parameters of  $q(x_i)$  with respect to the parameters of  $q(\theta_t)$ .  $q(x_i)$  can be computed efficiently using inside-outside. Thus, we can perform an EM-like alternation between calculating  $\hat{\alpha}$  and  $\hat{\theta}$ .<sup>4</sup>

It is also possible to estimate the hyper-parameters  $\alpha$  from data, a practice known as *empirical Bayes*, by optimizing  $\mathcal{F}$ . We explore learning separate hyper-parameters  $\alpha_t$  for each  $\theta_t$ , using a fixed point update described by Minka (2000), where  $k_t$  is the number of rules with parent state  $t$ :

$$\alpha'_t = \left( \frac{1}{\alpha_t} + \frac{1}{k_t \alpha_t^2} \left( \frac{\partial^2 \mathcal{F}}{\partial \alpha_t^2} \right)^{-1} \left( \frac{\partial \mathcal{F}}{\partial \alpha_t} \right) \right)^{-1}$$

## 7 Training and decoding

We implement our VB training algorithm inside the tree transducer package Tiburon (May and Knight, 2006), and experiment with both manually set and automatically estimated priors. For our manually set priors, we explore different hyper-parameter settings for three different priors, one for each of the main decision types: MR rule, NL pattern, and word generation. For the automatic priors, we estimate separate hyper-parameters for each multinomial (of which there are hundreds). As is standard, we initialize the word distributions using a variant of IBM model 1, and make use of NP lists (a manually created list of the constants in the MR language paired with the words that refer to them in the corpus).

At test time, since finding the most probable *MR* for a sentence involves summing over all possible derivations, we instead find the MR associated with the most probable *derivation*.

## 8 Experimental setup and evaluation

We evaluate the system on GeoQuery (Wong and Mooney, 2006), a parallel corpus of 880 English questions and database queries about United States geography, 250 of which were translated into Spanish, Japanese, and Turkish. We present here additional translations of the full 880 sentences into

<sup>4</sup>Because of the resemblance to EM, this procedure has been called VBEM. Unlike EM, however, this procedure alternates between two estimation steps and has no maximization step.

German, Greek, and Thai. For evaluation, following from Kwiatkowski et al. (2010), we reserve 280 sentences for test and train on the remaining 600. During development, we use cross-validation on the 600 sentence training set. At test, we run once on the remaining 280 and perform 10 fold cross-validation on the 250 sentence sets.

To judge correctness, we follow standard practice and submit each parse as a GeoQuery database query, and say the parse is correct only if the answer matches the gold standard. We report raw accuracy (the percentage of sentences with correct answers), as well as F1: the harmonic mean of precision (the proportion of correct answers out of sentences with a parse) and recall (the proportion of correct answers out of all sentences).<sup>5</sup>

We run three other state-of-the-art systems for comparison. *WASP* (Wong and Mooney, 2006) and the *hybrid tree* (Lu et al., 2008) are chosen to represent tree transformation based approaches, and, while this comparison is our primary focus, we also report *UBL-S* (Kwiatkowski et al., 2010) as a non-tree based top-performing system.<sup>6</sup> The hybrid tree is notable as the only other system based on a generative model, and *uni-hybrid*, a version that uses a unigram distribution over words, is very similar to our own model. We also report the best performing version, *re-hybrid*, which incorporates a discriminative re-ranking step.

We report transducer performance under three different training conditions: *tsEM* using EM, *tsVB-auto* using VB with empirical Bayes, and *tsVB-hand* using hyper-parameters manually tuned on the German training data ( $\alpha$  of 0.3, 0.8, and 0.25 for MR rule, NL pattern, and word choices, respectively).

Table 1 shows results for 10 fold cross-validation on the training set. The results highlight the benefit of the Dirichlet prior, whether manually or automatically set. VB improves over EM considerably, most likely because (1) the handling of unknown words and MR entities allows it to return an analysis for all sentences, and (2) the sparse Dirichlet prior favors fewer rules, reasonable in this setting where only a few words are likely to share the same meaning.

<sup>5</sup>Note that accuracy and f-score reduce to the same formula if there are no parse failures.

<sup>6</sup>UBL-S is based on CCG, which can be viewed as a mapping between graphs more general than trees.

DEV	geo600 - 10 fold cross-val			
	German		Greek	
	Acc	F1	Acc	F1
UBL-S	76.7	76.9	76.2	76.5
WASP	66.3	75.0	71.2	79.7
uni-hybrid	61.7	66.1	71.0	75.4
re-hybrid	62.3	69.5	70.2	76.8
tsEM	61.7	67.9	67.3	73.2
tsVB-auto	74.0	74.0	<b>•79.8</b>	<b>•79.8</b>
tsVB-hand	<b>•78.0</b>	<b>•78.0</b>	79.0	79.0
	English		Thai	
UBL-S	<b>85.3</b>	<b>85.4</b>	74.0	74.1
WASP	73.5	79.4	69.8	73.9
uni-hybrid	76.3	79.0	71.3	73.7
re-hybrid	77.0	82.2	71.7	76.0
tsEM	73.5	78.1	69.8	72.9
tsVB-auto	81.2	81.2	74.7	74.7
tsVB-hand	<b>•83.7</b>	<b>•83.7</b>	<b>•76.7</b>	<b>•76.7</b>

Table 1: Accuracy and F1 score comparisons on the geo600 training set. Highest scores are in bold, while the highest among the *tree based* models are marked with a bullet. The dotted line separates the tree based from non-tree based models.

On the test set (Table 2), we only run the model variants that perform best on the training set. Test set accuracy is consistently higher for the VB trained tree transducer than the other tree transformation based models (and often highest overall), while f-score remains competitive.<sup>7</sup>

## 9 Conclusion

We have argued that tree transformation based semantic parsing can benefit from the literature on formal language theory and tree automata, and have taken a step in this direction by presenting a tree transducer based semantic parser. Drawing this connection facilitates a greater flow of ideas in the research community, allowing semantic parsing to leverage ideas from other work with tree automata, while making clearer how seemingly isolated efforts might relate to one another. We demonstrate this by both building on previous work in training tree transducers using EM (Graehl et al., 2008),

<sup>7</sup>Numbers differ slightly here from previously published results due to the fact that we have standardized the inputs to the different systems.

TEST	geo880 - 600 train/280 test			
	German		Greek	
	Acc	F1	Acc	F1
UBL-S	<b>75.0</b>	<b>75.0</b>	73.6	73.7
WASP	65.7	<b>•74.9</b>	70.7	<b>•78.6</b>
re-hybrid	62.1	68.5	69.3	74.6
tsVB-hand	<b>•74.6</b>	74.6	<b>•75.4</b>	75.4
	English		Thai	
UBL-S	<b>82.1</b>	<b>82.1</b>	66.4	66.4
WASP	71.1	77.7	71.4	75.0
re-hybrid	76.8	<b>•81.0</b>	73.6	76.7
tsVB-hand	<b>•79.3</b>	79.3	<b>•78.2</b>	<b>•78.2</b>
	geo250 - 10 fold cross-val			
	English		Spanish	
UBL-S	80.4	80.6	79.7	80.1
WASP	70.0	80.8	72.4	81.0
re-hybrid	74.8	82.6	78.8	<b>•86.2</b>
tsVB-hand	<b>•83.2</b>	<b>•83.2</b>	<b>•80.0</b>	80.0
	Japanese		Turkish	
UBL-S	<b>80.5</b>	80.6	74.2	74.9
WASP	74.4	<b>•82.9</b>	62.4	75.9
re-hybrid	76.8	82.4	66.8	<b>•77.5</b>
tsVB-hand	<b>•78.0</b>	78.0	<b>•75.6</b>	75.6

Table 2: Accuracy and F1 score comparisons on the geo880 and geo250 test sets. Highest scores are in bold, while the highest among the *tree based* models are marked with a bullet. The dotted line separates the tree based from non-tree based models.<sup>7</sup>

and describing a general purpose variational inference algorithm for adapting tree transducers to the Bayesian framework. The new VB algorithm results in an overall performance improvement for the transducer over EM training, and the general effectiveness of the approach is further demonstrated by the Bayesian transducer achieving highest accuracy among other tree transformation based approaches.

## Acknowledgments

We thank Joel Lang, Michael Auli, Stella Frank, Prachya Boonkwan, Christos Christodoulopoulos, Ioannis Konstas, and Tom Kwiatkowski for providing the new translations of GeoQuery. This research was supported in part under the Australian Research Council’s Discovery Projects funding scheme (project number DP110102506).

## References

- Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- Benjamin Börschinger, Bevan K. Jones, and Mark Johnson. Reducing grounded learning tasks to grammatical inference. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, 2011.
- David Chiang, Jonathan Graehl, Kevin Knight, Adam Pauls, and Sujith Ravi. Bayesian inference for finite-state transducers. In *Proc. of the annual meeting of the North American Association for Computational Linguistics*, 2010.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What’s in a translation rule? In *Proc. of the annual meeting of the North American Association for Computational Linguistics*, 2004.
- Ruifang Ge and Raymond J. Mooney. A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Conference on Computational Natural Language Learning*, 2005.
- Jonathon Graehl, Kevin Knight, and Jon May. Training tree transducers. *Computational Linguistics*, 34:391–427, 2008.
- Rohit J. Kate and Raymond J. Mooney. Using string-kernels for learning semantic parsers. In *Proc. of the International Conference on Computational Linguistics and the annual meeting of the Association for Computational Linguistics*, 2006.
- Kevin Knight and Jonathon Graehl. An overview of probabilistic tree transducers for natural language processing. In *Proc. of the 6th International Conference on Intelligent Text Processing and Computational Linguistics*, 2005.
- Kenichi Kurihara and Taisuke Sato. Variational Bayesian grammar induction for natural language. In *Proc. of the 8th International Colloquium on Grammatical Inference*, 2006.
- Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Inducing probabilistic CCG grammars from logical form with higher-order unification. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, 2010.
- Percy Liang, Michael I. Jordan, and Dan Klein. Learning dependency-based compositional semantics. In *Proc. of the annual meeting of the Association for Computational Linguistics*, 2011.
- Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. A generative model for parsing natural language to meaning representations. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, 2008.
- Andreas Maletti, Jonathan Graehl, Mark Hopkins, and Kevin Knight. The power of extended top-down tree transducers. *SIAM J. Comput.*, 39:410–430, June 2009.
- Jon May and Kevin Knight. Tiburon: A weighted tree automata toolkit. In *Proc. of the International Conference on Implementation and Application of Automata*, 2006.
- Tom Minka. Estimating a Dirichlet distribution. Technical report, M.I.T., 2000.
- W.C. Rounds. Mappings and grammars on trees. *Mathematical Systems Theory* 4, pages 257–287, 1970.
- Stuart M. Shieber. Synchronous grammars as tree transducers. In *Proc. of the Seventh International Workshop on Tree Adjoining Grammar and Related Formalisms*, 2004.
- J.W. Thatcher. Generalized sequential machine maps. *J. Comput. System Sci.* 4, pages 339–367, 1970.
- Yuk Wah Wong and Raymond J. Mooney. Learning for semantic parsing with statistical machine translation. In *Proc. of Human Language Technology Conference and the annual meeting of the North American Chapter of the Association for Computational Linguistics*, 2006.
- Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proc. of the annual meeting of the Association for Computational Linguistics*, 2001.

# Dependency Hashing for $n$ -best CCG Parsing

Dominick Ng and James R. Curran

ϱ-lab, School of Information Technologies

University of Sydney

NSW, 2006, Australia

{dominick.ng, james.r.curran}@sydney.edu.au

## Abstract

Optimising for one grammatical representation, but evaluating over a different one is a particular challenge for parsers and  $n$ -best CCG parsing. We find that this mismatch causes many  $n$ -best CCG parses to be semantically equivalent, and describe a hashing technique that eliminates this problem, improving oracle  $n$ -best F-score by 0.7% and reranking accuracy by 0.4%. We also present a comprehensive analysis of errors made by the C&C CCG parser, providing the first breakdown of the impact of implementation decisions, such as supertagging, on parsing accuracy.

## 1 Introduction

Reranking techniques are commonly used for improving the accuracy of parsing (Charniak and Johnson, 2005). Efficient decoding of a parse forest is infeasible without dynamic programming, but this restricts features to local tree contexts. Reranking operates over a list of  $n$ -best parses according to the original model, allowing poor local parse decisions to be identified using arbitrary rich parse features.

The performance of reranking depends on the quality of the underlying  $n$ -best parses. Huang and Chiang (2005)'s  $n$ -best algorithms are used in a wide variety of parsers, including an  $n$ -best version of the C&C CCG parser (Clark and Curran, 2007; Brennan, 2008). The oracle F-score of this parser (calculated by selecting the most optimal parse in the  $n$ -best list) is 92.60% with  $n = 50$  over a baseline 1-best F-score of 86.84%. In contrast, the Charniak parser records an oracle F-score of 96.80% in 50-best mode

over a baseline of 91.00% (Charniak and Johnson, 2005). The 4.2% oracle score difference suggests that further optimisations may be possible for CCG.

We describe how  $n$ -best parsing algorithms that operate over derivations do not account for absorption ambiguities in parsing, causing semantically identical parses to exist in the CCG  $n$ -best list. This is caused by the mismatch between the optimisation target (different derivations) and the evaluation target (CCG dependencies). We develop a hashing technique over dependencies that removes duplicates and improves the oracle F-score by 0.7% to 93.32% and reranking accuracy by 0.4%. Huang et al. (2006) proposed a similar idea where strings generated by a syntax-based MT rescoring system were hashed to prevent duplicate translations.

Despite this improvement, there is still a substantial gap between the C&C and Charniak oracle F-scores. We perform a comprehensive subtractive analysis of the C&C parsing pipeline, identifying the relative contribution of each error class and why the gap exists. The parser scores 99.49% F-score with gold-standard categories on section 00 of CCGbank, and 94.32% F-score when returning the best parse in the chart using the supertagger on standard settings. Thus the supertagger contributes roughly 5% of parser error, and the parser model the remaining 7.5%. Various other speed optimisations also detrimentally affect accuracy to a smaller degree.

Several subtle trade-offs are made in parsers between speed and accuracy, but their actual impact is often unclear. Our work investigates these and the general issue of how different optimisation and evaluation targets can affect parsing performance.

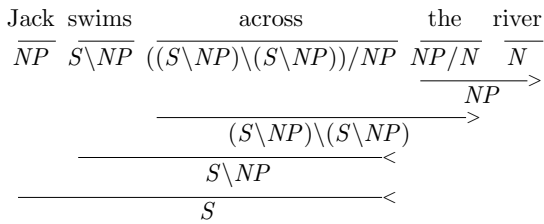


Figure 1: A CCG derivation with a PP adjunct, demonstrating forward and backward combinator application. Adapted from Villavicencio (2002).

## 2 Background

Combinatory Categorical Grammar (CCG, Steedman, 2000) is a lexicalised grammar formalism based on formal logic. The grammar is directly encoded in the lexicon in the form of *categories* that govern the syntactic behaviour of each word.

Atomic categories such as  $N$  (noun),  $NP$  (noun phrase), and  $PP$  (prepositional phrase) represent complete units. Complex categories encode subcategorisation information and are functors of the form  $X/Y$  or  $X \setminus Y$ . They represent structures which combine with an argument category  $Y$  to produce a result category  $X$ . In Figure 1, the complex category  $S \setminus NP$  for *swims* represents an intransitive verb requiring a subject  $NP$  to the left.

Combinatory rules are used to combine categories together to form an analysis. The simplest rules are forward and backward application, where complex categories combine with their outermost arguments. Forward and backward composition allow categories to be combined in a non-canonical order, and type-raising turns a category into a higher-order functor. A ternary coordination rule combines two identical categories separated by a *conj* into one.

As complex categories are combined with their arguments, they create a logical form representing the syntactic and semantic properties of the sentence. This logical form can be expressed in many ways; we will focus on the *dependency* representation used in CCGbank (Hockenmaier and Steedman, 2007). In Figure 1, *swims* generates one dependency:

$$\langle \text{swims}, S[\text{dcl}] \setminus NP_1, 1, \text{Jack}, - \rangle$$

where the dependency contains the head word, head category, argument slot, argument word, and whether the dependency is long-range.

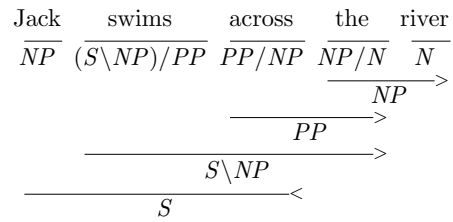


Figure 2: A CCG derivation with a PP argument (note the categories of *swims* and *across*). The bracketing is identical to Figure 1, but nearly all dependencies have changed.

## 2.1 Corpora and evaluation

CCGbank (Hockenmaier, 2003) is a transformation of the Penn Treebank (PTB) data into CCG derivations, and it is the standard corpus for English CCG parsing. Other CCG corpora have been induced in a similar way for German (Hockenmaier, 2006) and Chinese (Tse and Curran, 2010). CCGbank contains 99.44% of the sentences from the PTB, and several non-standard rules were necessary to achieve this coverage. These include punctuation absorption rules and unary type-changing rules for clausal adjuncts that are otherwise difficult to represent.

The standard CCG parsing evaluation calculates labeled precision, recall, and F-score over the dependencies recovered by a parser as compared to CCGbank (Clark et al., 2002). All components of a dependency must match the gold standard for it to be scored as correct, and this makes the procedure much harsher than the PARSEVAL labeled brackets metric. In Figure 2, the PP *across the river* has been interpreted as an argument rather than an adjunct as in Figure 1. Both parses would score identically under PARSEVAL as their bracketing is unchanged. However, the adjunct to argument change results in different categories for *swims* and *across*; nearly every CCG dependency in the sentence is headed by one of these two words and thus each one changes as a result. An incorrect argument/adjunct distinction in this sentence produces a score close to 0.

All experiments in this paper use the normal-form C&C parser model over CCGbank 00 (Clark and Curran, 2007). Scores are reported for sentences which the parser could analyse; we observed similar conclusions when repeating our experiments over the subset of sentences that were parsable under all configurations described in this paper.

## 2.2 The C&C parser

The C&C parser (Clark and Curran, 2007) is a fast and accurate CCG parser trained on CCGbank 02-21, with an accuracy of 86.84% on CCGbank 00 with the normal-form model. It is a two-phase system, where a supertagger assigns possible categories to words in a sentence and the parser combines them using the CKY algorithm. An  $n$ -best version incorporating the Huang and Chiang (2005) algorithms has been developed (Brennan, 2008). Recent work on a softmax-margin loss function and integrated supertagging via belief propagation has improved this to 88.58% (Auli and Lopez, 2011).

A parameter  $\beta$  is passed to the supertagger as a multi-tagging probability beam.  $\beta$  is initially set at a very restrictive value, and if the parser cannot form an analysis the supertagger is rerun with a lower  $\beta$ , returning more categories and giving the parser more options in constructing a parse. This *adaptive* supertagging prunes the search space whilst maintaining coverage of over 99%.

The supertagger also uses a *tag dictionary*, as described by Ratnaparkhi (1996), and accepts a cutoff  $k$ . Words seen more than  $k$  times in CCGbank 02-21 may only be assigned categories seen with that word more than 5 times in CCGbank 02-21; the frequency must also be no less than 1/500th of the most frequent tag for that word. Words seen fewer than  $k$  times may only be assigned categories seen with the POS of the word in CCGbank 02-21, subject to the cutoff and ratio constraint (Clark and Curran, 2004b). The tag dictionary eliminates infrequent categories and improves the performance of the supertagger, but at the cost of removing unseen or infrequently seen categories from consideration.

The parser accepts POS-tagged text as input; unlike many PTB parsers, these tags are fixed and remain unchanged throughout during the parsing pipeline. The POS tags are important features for the supertagger; parsing accuracy using gold-standard POS tags is typically 2% higher than using automatically assigned POS tags (Clark and Curran, 2004b).

## 2.3 $n$ -best parsing and reranking

Most parsers use dynamic programming, discarding infeasible states in order to maintain tractability. However, constructing an  $n$ -best list requires keep-

ing the top  $n$  states throughout. Huang and Chiang (2005) define several  $n$ -best algorithms that allow dynamic programming to be retained whilst generating precisely the top  $n$  parses – using the observation that once the 1-best parse is generated, the 2nd best parse must differ in exactly one location from it, and so forth. These algorithms are defined on a hypergraph framework equivalent to a chart, so the parses are distinguished based on their derivations. Huang et al. (2006) develop a translation reranking model using these  $n$ -best algorithms, but faced the issue of different derivations yielding the same string. This was overcome by storing a hashtable of strings at each node in the tree, and rejecting any derivations that yielded a previously seen string.

Collins (2000)’s parser reranker uses  $n$ -best parses of PTB 02-21 as training data. Reranker features include lexical heads and the distances between them, context-free rules in the tree,  $n$ -grams and their ancestors, and parent-grandparent relationships. The system improves the accuracy of the Collins parser from 88.20% to 89.75%.

Charniak and Johnson (2005)’s reranker uses a similar setup to the Collins reranker, but utilises much higher quality  $n$ -best parses. Additional features on top of those from the Collins reranker such as subject-verb agreement,  $n$ -gram local trees, and right-branching factors are also used. In 50-best mode the parser has an oracle F-score of 96.80%, and the reranker produces a final F-score of 91.00% (compared to an 89.70% baseline).

## 3 Ambiguity in $n$ -best CCG parsing

The type-raising and composition combinators allow the same logical form to be created from different category combination orders in a derivation. This is termed *spurious ambiguity*, where different derivational structures are semantically equivalent and will evaluate identically despite having a different phrase structure. The C&C parser employs the normal-form constraints of Eisner (1996) to address spurious ambiguity in 1-best parsing.

*Absorption ambiguity* occurs when a constituent may be legally placed at more than one location in a derivation, and all of the resulting derivations are semantically equivalent. Punctuation such as commas, brackets, and periods are particularly prone to

	Avg P/sent	Distinct P/sent	% Distinct
10-best	9.8	5.1	52
50-best	47.6	16.0	34
10-best <sup>#</sup>	9.0	9.0	100
50-best <sup>#</sup>	37.9	37.9	100

Table 1: Average and distinct parses per sentence over CCGbank 00 with respect to CCG dependencies. # indicates the inclusion of dependency hashing

absorption ambiguity in CCG; Figure 3 depicts four semantically equivalent sequences of absorption and combinator application in a sentence fragment.

The Brennan (2008) CCG  $n$ -best parser differentiates CCG parses by derivation rather than logical form. To illustrate how this is insufficient, we ran the parser using Algorithm 3 of Huang and Chiang (2005) with  $n = 10$  and  $n = 50$ , and calculated how many parses were semantically distinct (i.e. yield different dependencies). The results (summarised in Table 1) are striking: just 52% of 10-best parses and 34% of 50-best parses are distinct. We can also see that fewer than  $n$  parses are found on average for each sentence; this is mostly due to shorter sentences that may only receive one or two parses.

We perform the same diversity experiment using the DepBank-style grammatical relations (GRs, King et al., 2003; Briscoe and Carroll, 2006) output of the parser. GRs are generated via a dependency to GR mapping in the parser as well as a post-processing script to clean up common errors (Clark and Curran, 2007). GRs provide a more formalism-neutral comparison and abstract away from the raw CCG dependencies; for example, in Figures 1 and 2, the dependency from *swims* to *Jack* would be abstracted into (subj swims Jack) and thus would be identical in both parses. Hence, there are even fewer distinct parses in the GR results summarised in Table 2: 45% and 27% of 10-best and 50-best parses respectively yield unique GRs.

### 3.1 Dependency hashing

To address this problem of semantically equivalent  $n$ -best parses, we define a uniqueness constraint over all the  $n$ -best candidates:

**Constraint.** *At any point in the derivation, any  $n$ -best candidate must not have the same dependencies as any candidate already in the list.*

	Avg P/sent	Distinct P/sent	% Distinct
10-best	9.8	4.4	45
50-best	47.6	13.0	27
10-best <sup>#</sup>	8.9	8.1	91
50-best <sup>#</sup>	37.1	31.5	85

Table 2: Average and distinct parses per sentence over CCGbank 00 with respect to GRs. # indicates the inclusion of dependency hashing

Enforcing this constraint is non-trivial as it is infeasible to directly compare every dependency in a partial tree with another. Due to the flexible notion of constituency in CCG, dependencies can be generated at a variety of locations in a derivation and in a variety of orders. This means that comparing all of the dependencies in a particular state may require traversing the entire sub-derivation at that point. Parsing is already a computationally expensive process, so we require as little overhead from this check as possible.

Instead, we represent all of the CCG dependencies in a sub-derivation using a hash value. This allows us to compare the dependencies in two derivations with a single numeric equality check rather than a full iteration. The underlying idea is similar to that of Huang et al. (2006), who maintain a hashtable of unique strings produced by a translation reranker, and reject new strings that have previously been generated. Our technique does not use a hashtable, and instead only stores the hash value for each set of dependencies, which is much more efficient but runs the risk of filtering unique parses due to collisions.

As we combine partial trees to build the derivation, we need to convolve the hash values in a consistent manner. The convolution operator must be *order-independent* as dependencies may be generated in an arbitrary order at different locations in each tree. We use the bitwise exclusive OR ( $\oplus$ ) operation as our convolution operator: when two partial derivations are combined, their hash values are XOR’ed together. XOR is commonly employed in hashing applications for randomly permuting numbers, and it is also order independent:  $a \oplus b \equiv b \oplus a$ . Using XOR, we enforce a unique hash value constraint in the  $n$ -best list of candidates, discarding potential candidates with an identical hash value to any already in the list.



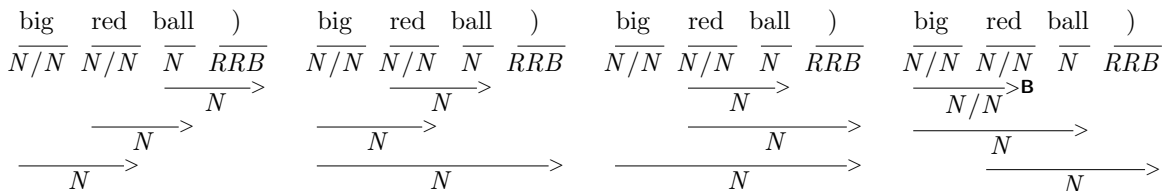


Figure 3: All four derivations have a different syntactic structure, but generate identical dependencies.

	Collisions	Comparisons	%
10-best	300	54861	0.55
50-best	2109	225970	0.93

Table 3: Dependency hash collisions and comparisons over 00 of CCGbank.

### 3.2 Hashing performance

We evaluate our hashing technique with several experiments. A simple test is to measure the number of collisions that occur, i.e. where two partial trees with different dependencies have the same hash value. We parsed CCGbank 00 with  $n = 10$  and  $n = 50$  using a 32 bit hash, and exhaustively checked the dependencies of colliding states. We found that less than 1% of comparisons resulted in collisions in both 10-best and 50-best mode, and decided that this was acceptably low for distinguishing duplicates.

We reran the diversity experiments, and verified that every  $n$ -best parse for every sentence in CCGbank 00 was unique (see Table 1), corroborating our decision to use hashing alone. On average, there are fewer parses per sentence, showing that hashing is eliminating many equivalent parses for more ambiguous sentences. However, hashing also leads to a near doubling of unique parses in 10-best mode and a 2.3x increase in 50-best mode. Similar results are recorded for the GR diversity (see Table 2), though not every set of GRs is unique due to the many-to-many mapping from CCG dependencies. These results show that hashing prunes away equivalent parses, creating more diversity in the  $n$ -best list.

We also evaluate the oracle F-score of the parser using dependency hashing. Our results in Table 4 include a 1.1% increase in 10-best mode and 0.72% in 50-best mode using the new constraints, showing how the diversified parse list contains better candidates for reranking. Our highest oracle F-score was 93.32% in 50-best mode.

Experiment	LP	LR	LF	AF
baseline	87.27	86.41	86.84	84.91
oracle 10-best	91.50	90.49	90.99	89.01
oracle 50-best	93.17	92.04	92.60	90.68
<b>oracle 10-best<sup>#</sup></b>	<b>92.67</b>	<b>91.51</b>	<b>92.09</b>	<b>90.15</b>
<b>oracle 50-best<sup>#</sup></b>	<b>94.00</b>	<b>92.66</b>	<b>93.32</b>	<b>91.47</b>

Table 4: Oracle precision, recall, and F-score on gold and auto POS tags for the C&C  $n$ -best parser. # denotes the inclusion of dependency hashing.

	Test data	
Training data	no hashing	hashing
no hashing	86.83	86.35
hashing	<b>87.21</b>	<b>87.15</b>

Table 5: Reranked parser accuracy; labeled F-score using gold POS tags, with and without dependency hashing

### 3.3 CCG reranking performance

Finally, we implement a discriminative maximum entropy reranker for the  $n$ -best C&C parser and evaluate it when using dependency hashing. We reimplement the features described in Charniak and Johnson (2005) and add additional features based on those used in the C&C parser and on features of CCG dependencies. The training data is cross-fold  $n$ -best parsed sentences of CCGbank 02-21, and we use the MEGAM optimiser<sup>1</sup> in regression mode to predict the labeled F-score of each  $n$ -best candidate parse.

Our experiments rerank the top 10-best parses and use four configurations: with and without dependency hashing for generating the training and test data for the reranker. Table 5 shows that labeled F-score improves substantially when dependency hashing is used to create reranker training data. There is a 0.4% improvement using no hashing at test, and a 0.8% improvement using hashing

<sup>1</sup><http://hal3.name/megam>

at test, showing that more diverse training data creates a better reranker. The results of 87.21% without hashing at test and 87.15% using hashing at test are statistically indistinguishable from one other; though we would expect the latter to perform better.

Our results also show that the reranker performs extremely poorly using diversified test parses and undiversified training parses. There is a 0.5% performance loss in this configuration, from 86.83% to 86.35% F-score. This may be caused by the reranker becoming attuned to selecting between semantically indistinguishable derivations, which are pruned away in the diversified test set.

## 4 Analysing parser errors

A substantial gap exists between the oracle F-score of our improved  $n$ -best parser and other PTB  $n$ -best parsers (Charniak and Johnson, 2005). Due to the different evaluation schemes, it is difficult to directly compare these numbers, but whether there is further room for improvement in CCG  $n$ -best parsing is an open question. We analyse three main classes of errors in the C&C parser in order to answer this question: grammar error, supertagger error, and model error. Furthermore, insights from this analysis will prove useful in evaluating tradeoffs made in parsers.

*Grammar error:* the parser implements a subset of the grammar and unary type-changing rules in CCGbank for efficiency, with some rules, such as substitution, omitted for efficiency (Clark and Curran, 2007). This means that, given the correct categories for words in a sentence, the parser may be unable to combine them into a derivation yielding the correct dependencies, or it may not recognise the gold standard category at all.

There is an additional constraint in the parser that only allows two categories to combine if they have been seen to combine in the training data. This *seen rules* constraint is used to reduce the size of the chart and improve parsing speed, at the cost of only permitting category combinations seen in CCGbank 02-21 (Clark and Curran, 2007).

*Supertagger error:* The supertagger uses a restricted set of 425 categories determined by a frequency cutoff of 10 over the training data (Clark and Curran, 2004b). Words with gold categories that are not in this set cannot be tagged correctly.

The  $\beta$  parameter restricts the categories to within a probability beam, and the tag dictionary restricts the set of categories that can be considered for each word. Supertagger model error occurs when the supertagger can assign a word its correct category, but the statistical model does not assign the correct tag enough probability for it to fall within the  $\beta$ .

*Model error:* The parser model features may be rich enough to capture certain characteristics of parses, causing it to select a suboptimal parse.

### 4.1 Subtractive experiments

We develop an oracle methodology to distinguish between grammar, supertagger, and model errors. This is the most comprehensive error analysis of a parsing pipeline in the literature.

First, we supplied gold-standard categories for each word in the sentence. In this experiment the parser only needs to combine the categories correctly to form the gold parse. In our testing over CCGbank 00, the parser scores 99.49% F-score given perfect categories, with 95.61% coverage. Thus, grammar error accounts for about 0.5% of overall parser errors as well as a 4.4% drop in coverage<sup>2</sup>. All results in this section will be compared against this 99.49% result as it removes the grammar error from consideration.

### 4.2 Supertagger and model error

To determine supertagger and model error, we run the parser on standard settings over CCGbank 00 and examined the chart. If it contains the gold parse, then a model error results if the parser returns any other parse. Otherwise, it is a supertagger or grammar error, where the parser cannot construct the best parse. For each sentence, we found the best parse in the chart by decoding against the gold dependencies. Each partial tree was scored using the formula:

$$score = n_{correct} - nbad$$

where  $n_{correct}$  is the number of dependencies which appear in the gold standard, and  $nbad$  is the number of dependencies which do not appear in the gold standard. The top scoring derivation in the tree under this scheme is then returned.

<sup>2</sup>Clark and Curran (2004a) performed a similar experiment with lower accuracy and coverage; our improved numbers are due to changes in the parser.

Experiment	LP	LR	LF	AF	cover	$\Delta$ LF	$\Delta$ AF
oracle cats	99.72	99.27	99.49	99.49	95.61	0.00	0.00
best in chart -tagdict -seen rules	96.88	94.81	95.84	94.17	99.01	-3.65	-5.32
best in chart -tagdict	96.13	94.72	95.42	93.56	99.37	-4.07	-5.93
best in chart -seen rules	96.10	93.66	94.86	93.35	98.85	-4.63	-6.14
best in chart	95.15	93.50	94.32	92.60	99.16	-5.17	-6.89
baseline	87.27	86.41	86.84	84.91	99.16	-12.65	-14.58

Table 6: Oracle labeled precision, recall, F-score, F-score with auto POS, and coverage over CCGbank 00. -tagdict indicates disabling the tag dictionary, -seen rules indicates disabling the seen rules constraint

$\beta$	$k$	cats/word	sent/sec	LP	LR	LF	AF	cover	$\Delta$ LF	$\Delta$ AF
gold cats		-	-	99.72	99.27	99.49	-	95.61	0.00	0.00
0.075	20	1.27	40.5	95.46	93.90	94.68	93.07	94.30	-4.81	-6.42
0.03	20	1.43	33.0	96.23	94.87	95.54	94.01	96.03	-3.95	-5.48
0.01	20	1.72	19.1	97.02	95.82	96.42	95.02	96.86	-3.07	-4.47
0.005	20	1.98	10.7	97.26	96.09	96.68	95.32	97.23	-2.81	-4.17
0.001	150	3.57	1.18	98.33	97.37	97.85	96.76	96.13	-1.64	-2.73

Table 7: Category ambiguity, speed, labeled P, R, F-score on gold and auto POS, and coverage over CCGbank 00 for the standard supertagger parameters selecting the best scoring parse against the gold parse in the chart.

We obtain an overall maximum possible F-score for the parser using this scoring formula. The difference between this maximum F-score and the oracle result of 99.49% represents supertagger error (where the supertagger has not provided the correct categories), and the difference to the baseline performance indicates model error (where the parser model has not selected the optimal parse given the current categories). We also try disabling the seen rules constraint to determine its impact on accuracy.

The impact of tag dictionary errors must be neutralised in order to distinguish between the types of supertagger error. To do this, we added the gold category for a word to the set of possible tags considered for that word by the supertagger. This was done for categories that the supertagger could use; categories that were not in the permissible set of 425 categories were not considered. This is an optimistic experiment; removing the tag dictionary entirely would greatly increase the number of categories considered by the supertagger and may dramatically change the tagging results.

Table 6 shows the results of our experiments. The delta columns indicate the difference in labeled F-score to the oracle result, which discounts the grammar error in the parser. We ran the experiment in four configurations: disabling the tag dictionary, dis-

abling the seen rules constraint, and disabling both. There are coverage differences of less than 0.5% that will have a small impact on these results.

The “best in chart” experiment produces a result of 94.32% with gold POS tags and 92.60% with auto POS tags. These numbers are the upper bound of the parser with the supertagger on standard settings. Our result with gold POS tags is statistically identical to the oracle experiment conducted by Auli and Lopez (2011), which exchanged brackets for dependencies in the forest oracle algorithm of Huang (2008). This illustrates the validity of our technique.

A perfect tag dictionary that always contains the gold standard category if it is available results in an upper bound accuracy of 95.42%. This shows that overall supertagger error in the parser is around 5.2%, with roughly 1% attributable to the use of the tag dictionary and the remainder to the supertagger model. The baseline parser is 12.5% worse than the oracle categories result due to model error and supertagger error, so model error accounts for roughly 7.3% of the loss.

Eliminating the seen rules constraint contributes to a 0.5% accuracy improvement over both the standard parser configuration and the -tagdict configuration, at the cost of roughly 0.3% coverage to both. This is of similar magnitude to grammar error; but

Experiment	LF	cover	$\Delta$ LF
baseline	86.84	99.16	0.00
auto POS parser	86.57	99.16	-0.27
auto POS super	85.33	99.06	-1.51
auto POS both	84.91	99.06	-1.93

Table 8: Labeled F-score, coverage, and deltas over CCGbank 00 for combinations of gold and auto POS tags.

here accuracy is traded off against coverage.

The results also show that model and supertagger error largely accounts for the remaining oracle accuracy difference between the C&C  $n$ -best parser and the Charniak/Collins  $n$ -best parsers. The absolute upper bound of the C&C parser is only 1% higher than the oracle 50-best score in Table 4, placing the  $n$ -best parser close to its theoretical limit.

### 4.3 Varying supertagger parameters

We conduct a further experiment to determine the impact of the standard  $\beta$  and  $k$  values used in the parser. We reran the “best in chart” configuration, but used each standard  $\beta$  and  $k$  value individually rather than backing off to a lower  $\beta$  value to find the maximum score at each individual value.

Table 7 shows that the oracle accuracy improves from 94.68% F-score and 94.30% coverage with  $\beta = 0.075, k = 20$  to 97.85% F-score and 96.13% coverage with  $\beta = 0.001, k = 150$ . At higher  $\beta$  values, accuracy is lost because the correct category is not returned to the parser, while lower  $\beta$  values are more likely to return the correct category. The coverage peaks at the second-lowest value because at lower  $\beta$  values, the number of categories returned means all of the possible derivations cannot be stored in the chart. The back-off approach substantially increases coverage by ensuring that parses that fail at higher  $\beta$  values are retried at lower ones, at the cost of reducing the upper accuracy bound to below that of any individual  $\beta$ .

The speed of the parser varies substantially in this experiment, from 40.5 sents/sec at the first  $\beta$  level to just 1.18 sents/sec at the last. This illustrates the trade-off in using supertagging: the maximum achievable accuracy drops by nearly 5% for parsing speeds that are an order of magnitude faster.

### 4.4 Gold and automatic POS tags

There is a substantial difference in accuracy between experiments that use gold POS and auto POS tags. Table 6 shows a corresponding drop in upper bound accuracy from 94.32% with gold POS tags to 92.60% with auto POS tags. Both the supertagger and parser use POS tags independently as features, but this result suggests that the bulk of the performance difference comes from the supertagger. To fully identify the error contributions, we ran an experiment where we provide gold POS tags to one of the parser and supertagger, and auto POS tags to the other, and then run the standard evaluation (the oracle experiment will be identical to the “best in chart”).

Table 8 shows that supplying the parser with auto POS tags reduces accuracy by 0.27% compared to the baseline parser, while supplying the supertagger with auto POS tags results in a 1.51% decrease. The parser uses more features in a wider context than the supertagger, so it is less affected by POS tag errors.

## 5 Conclusion

We have described how a mismatch between the way CCG parses are modeled and evaluated caused equivalent parses to be produced in  $n$ -best parsing. We eliminate duplicates by hashing dependencies, significantly improving the oracle F-score of CCG  $n$ -best parsing by 0.7% to 93.32%, and improving the performance of CCG reranking by up to 0.4%.

We have comprehensively investigated the sources of error in the C&C parser to explain the gap in oracle performance compared with other  $n$ -best parsers. We show the impact of techniques that subtly trade off accuracy for speed and coverage. This will allow a better choice of parameters for future applications of parsing in CCG and other lexicalised formalisms.

### Acknowledgments

We would like to thank the reviewers for their comments. This work was supported by Australian Research Council Discovery grant DP1097291, the Capital Markets CRC, an Australian Postgraduate Award, and a University of Sydney Vice-Chancellor’s Research Scholarship.

## References

- Michael Auli and Adam Lopez. 2011. Training a Log-Linear Parser with Loss Functions via Softmax-Margin. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 333–343. Edinburgh, Scotland, UK.
- Forrest Brennan. 2008. *k-best Parsing Algorithms for a Natural Language Parser*. Master’s thesis, University of Oxford.
- Ted Briscoe and John Carroll. 2006. Evaluating the Accuracy of an Unlexicalized Statistical Parser on the PARC DepBank. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 41–48. Sydney, Australia.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-Fine n-Best Parsing and MaxEnt Discriminative Reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pages 173–180. Ann Arbor, Michigan, USA.
- Stephen Clark and James R. Curran. 2004a. Parsing the WSJ Using CCG and Log-Linear Models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 103–110. Barcelona, Spain.
- Stephen Clark and James R. Curran. 2004b. The Importance of Supertagging for Wide-Coverage CCG Parsing. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04)*, pages 282–288. Geneva, Switzerland.
- Stephen Clark and James R. Curran. 2007. Wide-Coverage Efficient Statistical Parsing with CCG and Log-Linear Models. *Computational Linguistics*, 33(4):493–552.
- Stephen Clark, Julia Hockenmaier, and Mark Steedman. 2002. Building Deep Dependency Structures using a Wide-Coverage CCG Parser. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-02)*, pages 327–334. Philadelphia, Pennsylvania, USA.
- Michael Collins. 2000. Discriminative Reranking for Natural Language Parsing. In *Proceedings of the 17th International Conference on Machine Learning (ICML-00)*, pages 175–182. Palo Alto, California, USA.
- Jason Eisner. 1996. Efficient Normal-Form Parsing for Combinatory Categorical Grammar. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, pages 79–86. Santa Cruz, California, USA.
- Julia Hockenmaier. 2003. Parsing with Generative Models of Predicate-Argument Structure. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 359–366. Sapporo, Japan.
- Julia Hockenmaier. 2006. Creating a CCGbank and a Wide-Coverage CCG Lexicon for German. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, pages 505–512. Sydney, Australia.
- Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.
- Liang Huang. 2008. Forest Reranking: Discriminative Parsing with Non-Local Features. In *Proceedings of the Human Language Technology Conference at the 45th Annual Meeting of the Association for Computational Linguistics (HLT/ACL-08)*, pages 586–594. Columbus, Ohio.
- Liang Huang and David Chiang. 2005. Better k-best Parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology (IWPT-05)*, pages 53–64. Vancouver, British Columbia, Canada.
- Liang Huang, Kevin Knight, and Aravind K. Joshi. 2006. Statistical Syntax-Directed Translation with Extended Domain of Locality. In *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA-06)*, pages 66–73. Boston, Massachusetts, USA.
- Tracy Holloway King, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. 2003. The PARC 700 Dependency Bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora*, pages 1–8. Budapest, Hungary.
- Adwait Ratnaparkhi. 1996. A Maximum Entropy Model for Part-of-Speech Tagging. In *Proceedings of the 1996 Conference on Empirical Methods in Natural Language Processing (EMNLP-96)*, pages 133–142. Philadelphia, Pennsylvania, USA.
- Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, Massachusetts, USA.
- Daniel Tse and James R. Curran. 2010. Chinese CCGbank: extracting CCG derivations from the Penn Chinese Treebank. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING-2010)*, pages 1083–1091. Beijing, China.
- Aline Villavicencio. 2002. Learning to Distinguish PP Arguments from Adjuncts. In *Proceedings of the 6th Conference on Natural Language Learning (CoNLL-2002)*, pages 84–90. Taipei, Taiwan.

# Strong Lexicalization of Tree Adjoining Grammars

Andreas Maletti\*

IMS, Universität Stuttgart  
Pfaffenwaldring 5b  
70569 Stuttgart, Germany  
maletti@ims.uni-stuttgart.de

Joost Engelfriet

LIACS, Leiden University  
P.O. Box 9512  
2300 RA Leiden, The Netherlands  
engelfri@liacs.nl

## Abstract

Recently, it was shown (KUHLMANN, SATTÀ: *Tree-adjoining grammars are not closed under strong lexicalization*. *Comput. Linguist.*, 2012) that finitely ambiguous tree adjoining grammars cannot be transformed into a normal form (preserving the generated tree language), in which each production contains a lexical symbol. A more powerful model, the simple context-free tree grammar, admits such a normal form. It can be effectively constructed and the maximal rank of the non-terminals only increases by 1. Thus, simple context-free tree grammars strongly lexicalize tree adjoining grammars and themselves.

## 1 Introduction

Tree adjoining grammars [TAG] (Joshi et al., 1969; Joshi et al., 1975) are a mildly context-sensitive grammar formalism that can handle certain non-local dependencies (Kuhlmann and Mohl, 2006), which occur in several natural languages. A good overview on TAG, their formal properties, their linguistic motivation, and their applications is presented by Joshi and Schabes (1992) and Joshi and Schabes (1997), in which also strong lexicalization is discussed. In general, lexicalization is the process of transforming a grammar into an equivalent one (potentially expressed in another formalism) such that each production contains a lexical item (or anchor). Each production can then be viewed as lexical information on its anchor. It demonstrates a syntactical construction in which the anchor can occur. Since a lexical item is a letter of the string

alphabet, each production of a lexicalized grammar produces at least one letter of the generated string. Consequently, lexicalized grammars offer significant parsing benefits (Schabes et al., 1988) as the number of applications of productions (i.e., derivation steps) is clearly bounded by the length of the input string. In addition, the lexical items in the productions guide the production selection in a derivation, which works especially well in scenarios with large alphabets.<sup>1</sup> The GREIBACH normal form (Hopcroft et al., 2001; Blum and Koch, 1999) offers those benefits for context-free grammars [CFG], but it changes the parse trees. Thus, we distinguish between two notions of equivalence: *Weak equivalence* (Bar-Hillel et al., 1960) only requires that the generated string languages coincide, whereas *strong equivalence* (Chomsky, 1963) requires that even the generated tree languages coincide. Correspondingly, we obtain weak and strong lexicalization based on the required equivalence.

The GREIBACH normal form shows that CFG can weakly lexicalize themselves, but they cannot strongly lexicalize themselves (Schabes, 1990). It is a prominent feature of tree adjoining grammars that they can strongly lexicalize CFG (Schabes, 1990),<sup>2</sup> and it was claimed and widely believed that they can strongly lexicalize themselves. Recently, Kuhlmann and Satta (2012) proved that TAG actually cannot strongly lexicalize themselves. In fact, they prove that TAG cannot even strongly lexicalize the weaker tree insertion grammars (Schabes and Waters, 1995). However, TAG can weakly lexicalize themselves (Fujiyoshi, 2005).

<sup>1</sup>Chen (2001) presents a detailed account.

<sup>2</sup>Good algorithmic properties and the good coverage of linguistic phenomena are other prominent features.

\* Financially supported by the German Research Foundation (DFG) grant MA 4959/1-1.

Simple (i.e., linear and nondeleting) context-free tree grammars [CFTG] (Rounds, 1969; Rounds, 1970) are a more powerful grammar formalism than TAG (Mönnich, 1997). However, the monadic variant is strongly equivalent to a slightly extended version of TAG, which is called non-strict TAG (Kepser and Rogers, 2011). A GREIBACH normal form for a superclass of CFTG (viz., second-order abstract categorical grammars) was discussed by Kanazawa and Yoshinaka (2005) and Yoshinaka (2006). In particular, they also demonstrate that monadic CFTG can strongly lexicalize regular tree grammars (Gécseg and Steinby, 1984; Gécseg and Steinby, 1997).

CFTG are weakly equivalent to the simple macro grammars of Fischer (1968), which are a notational variant of the well-nested linear context-free rewriting systems (LCFRS) of Vijay-Shanker et al. (1987) and the well-nested multiple context-free grammars (MCFG) of Seki et al. (1991).<sup>3</sup> Thus, CFTG are mildly context-sensitive since their generated string languages are semi-linear and can be parsed in polynomial time (Gómez-Rodríguez et al., 2010).

In this contribution, we show that CFTG can strongly lexicalize TAG and also themselves, thus answering the second question in the conclusion of Kuhlmann and Satta (2012). This is achieved by a series of normalization steps (see Section 4) and a final lexicalization step (see Section 5), in which a lexical item is guessed for each production that does not already contain one. This item is then transported in an additional argument until it is exchanged for the same item in a terminal production. The lexicalization is effective and increases the maximal rank (number of arguments) of the non-terminals by at most 1. In contrast to a transformation into GREIBACH normal form, our lexicalization does not radically change the structure of the derivations. Overall, our result shows that if we consider only lexicalization, then CFTG are a more natural generalization of CFG than TAG.

## 2 Notation

We write  $[k]$  for the set  $\{i \in \mathbb{N} \mid 1 \leq i \leq k\}$ , where  $\mathbb{N}$  denotes the set of nonnegative integers. We use a fixed countably infinite set  $X = \{x_1, x_2, \dots\}$

<sup>3</sup>Kuhlmann (2010), Mönnich (2010), and Kanazawa (2009) discuss well-nestedness.

of (mutually distinguishable) variables, and we let  $X_k = \{x_i \mid i \in [k]\}$  be the first  $k$  variables from  $X$  for every  $k \in \mathbb{N}$ . As usual, an alphabet  $\Sigma$  is a finite set of symbols, and a ranked alphabet  $(\Sigma, \text{rk})$  adds a ranking  $\text{rk}: \Sigma \rightarrow \mathbb{N}$ . We let  $\Sigma_k = \{\sigma \mid \text{rk}(\sigma) = k\}$  be the set of  $k$ -ary symbols. Moreover, we just write  $\Sigma$  for the ranked alphabet  $(\Sigma, \text{rk})$ .<sup>4</sup> We build trees over the ranked alphabet  $\Sigma$  such that the nodes are labeled by elements of  $\Sigma$  and the rank of the node label determines the number of its children. In addition, elements of  $X$  can label leaves. Formally, the set  $T_\Sigma(X)$  of  $\Sigma$ -trees indexed by  $X$  is the smallest set  $T$  such that  $X \subseteq T$  and  $\sigma(t_1, \dots, t_k) \in T$  for all  $k \in \mathbb{N}$ ,  $\sigma \in \Sigma_k$ , and  $t_1, \dots, t_k \in T$ .<sup>5</sup>

We use positions to address the nodes of a tree. A position is a sequence of nonnegative integers indicating successively in which subtree the addressed node is. More precisely, the root is at position  $\varepsilon$  and the position  $ip$  with  $i \in \mathbb{N}$  and  $p \in \mathbb{N}^*$  refers to the position  $p$  in the  $i^{\text{th}}$  direct subtree. Formally, the set  $\text{pos}(t) \subseteq \mathbb{N}^*$  of positions of a tree  $t \in T_\Sigma(X)$  is defined by  $\text{pos}(x) = \{\varepsilon\}$  for  $x \in X$  and

$$\text{pos}(\sigma(t_1, \dots, t_k)) = \{\varepsilon\} \cup \{ip \mid i \in [k], p \in \text{pos}(t_i)\}$$

for all symbols  $\sigma \in \Sigma_k$  and  $t_1, \dots, t_k \in T_\Sigma(X)$ . The positions are indicated as superscripts of the labels in the tree of Figure 1. The subtree of  $t$  at position  $p \in \text{pos}(t)$  is denoted by  $t|_p$ , and the label of  $t$  at position  $p$  by  $t(p)$ . Moreover,  $t[u]_p$  denotes the tree obtained from  $t$  by replacing the subtree at  $p$  by the tree  $u \in T_\Sigma(X)$ . For every label set  $S \subseteq \Sigma$ , we let  $\text{pos}_S(t) = \{p \in \text{pos}(t) \mid t(p) \in S\}$  be the  $S$ -labeled positions of  $t$ . For every  $\sigma \in \Sigma$ , we let  $\text{pos}_\sigma(t) = \text{pos}_{\{\sigma\}}(t)$ . The set  $C_\Sigma(X_k)$  contains all trees  $t$  of  $T_\Sigma(X)$ , in which every  $x \in X_k$  occurs exactly once and  $\text{pos}_{X \setminus X_k}(t) = \emptyset$ . Given  $u_1, \dots, u_k \in T_\Sigma(X)$ , the first-order substitution  $t[u_1, \dots, u_k]$  is inductively defined by

$$x_i[u_1, \dots, u_k] = \begin{cases} u_i & \text{if } i \in [k] \\ x_i & \text{otherwise} \end{cases}$$

$$t[u_1, \dots, u_k] = \sigma(t_1[u_1, \dots, u_k], \dots, t_k[u_1, \dots, u_k])$$

for every  $i \in \mathbb{N}$  and  $t = \sigma(t_1, \dots, t_k)$  with  $\sigma \in \Sigma_k$  and  $t_1, \dots, t_k \in T_\Sigma(X)$ . First-order substitution is illustrated in Figure 1.

<sup>4</sup>We often decorate a symbol  $\sigma$  with its rank  $k$  [e.g.  $\sigma^{(k)}$ ].

<sup>5</sup>We will often drop quantifications like ‘for all  $k \in \mathbb{N}$ ’.

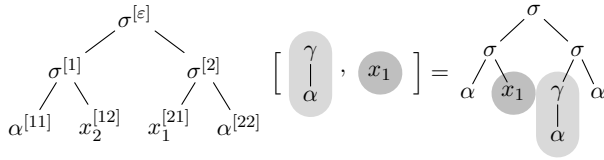


Figure 1: Tree in  $C_\Sigma(X_2) \subset T_\Sigma(X)$  with indicated positions, where  $\Sigma = \{\sigma, \gamma, \alpha\}$  with  $\text{rk}(\sigma) = 2$ ,  $\text{rk}(\gamma) = 1$ , and  $\text{rk}(\alpha) = 0$ , and an example first-order substitution.

In first-order substitution we replace leaves (elements of  $X$ ), whereas in second-order substitution we replace an internal node (labeled by a symbol of  $\Sigma$ ). Let  $p \in \text{pos}(t)$  be such that  $t(p) \in \Sigma_k$ , and let  $u \in C_\Sigma(X_k)$  be a tree in which the variables  $X_k$  occur exactly once. The second-order substitution  $t[p \leftarrow u]$  replaces the subtree at position  $p$  by the tree  $u$  into which the children of  $p$  are (first-order) substituted. In essence,  $u$  is “folded” into  $t$  at position  $p$ . Formally,  $t[p \leftarrow u] = t[u[t|_1, \dots, t|_k]]_p$ . Given  $P \subseteq \text{pos}_\sigma(t)$  with  $\sigma \in \Sigma_k$ , we let  $t[P \leftarrow u]$  be  $t[p_1 \leftarrow u] \dots [p_n \leftarrow u]$ , where  $P = \{p_1, \dots, p_n\}$  and  $p_1 > \dots > p_n$  in the lexicographic order. Second-order substitution is illustrated in Figure 2. Gécseg and Steinby (1997) present a detailed introduction to trees and tree languages.

### 3 Context-free tree grammars

In this section, we recall linear and nondeleting context-free tree grammars [CFTG] (Rounds, 1969; Rounds, 1970). The property ‘linear and nondeleting’ is often called ‘simple’. The nonterminals of regular tree grammars only occur at the leaves and are replaced using first-order substitution. In contrast, the nonterminals of a CFTG are ranked symbols, can occur anywhere in a tree, and are replaced using second-order substitution.<sup>6</sup> Consequently, the nonterminals  $N$  of a CFTG form a ranked alphabet. In the left-hand sides of productions we write  $A(x_1, \dots, x_k)$  for a nonterminal  $A \in N_k$  to indicate the variables that hold the direct subtrees of a particular occurrence of  $A$ .

**Definition 1.** A (simple) context-free tree grammar [CFTG] is a system  $(N, \Sigma, S, P)$  such that

- $N$  is a ranked alphabet of *nonterminal symbols*,
- $\Sigma$  is a ranked alphabet of *terminal symbols*,<sup>7</sup>

<sup>6</sup>see Sections 6 and 15 of (Gécseg and Steinby, 1997)

<sup>7</sup>We assume that  $\Sigma \cap N = \emptyset$ .

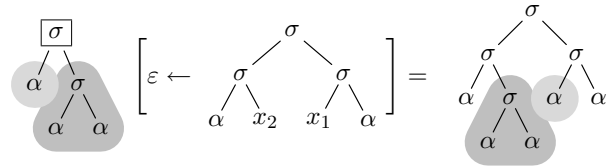


Figure 2: Example second-order substitution, in which the boxed symbol  $\sigma$  is replaced.

- $S \in N_0$  is the *start nonterminal* of rank 0, and
- $P$  is a finite set of *productions* of the form  $A(x_1, \dots, x_k) \rightarrow r$ , where  $r \in C_{N \cup \Sigma}(X_k)$  and  $A \in N_k$ .

The components  $\ell$  and  $r$  are called left- and right-hand side of the production  $\ell \rightarrow r$  in  $P$ . We say that it is an  $A$ -production if  $\ell = A(x_1, \dots, x_k)$ . The right-hand side is simply a tree using terminal and nonterminal symbols according to their rank. Moreover, it contains all the variables of  $X_k$  exactly once. Let us illustrate the syntax on an example CFTG. We use an abstract language for simplicity and clarity. We use lower-case Greek letters for terminal symbols and upper-case Latin letters for nonterminals.

**Example 2.** As a running example, we consider the CFTG  $G_{\text{ex}} = (\{S^{(0)}, A^{(2)}\}, \Sigma, S, P)$  where

- $\Sigma = \{\sigma^{(2)}, \alpha^{(0)}, \beta^{(0)}\}$  and
- $P$  contains the productions (see Figure 3):<sup>8</sup>

$$S \rightarrow A(\alpha, \alpha) \mid A(\beta, \beta) \mid \sigma(\alpha, \beta)$$

$$A(x_1, x_2) \rightarrow A(\sigma(x_1, S), \sigma(x_2, S)) \mid \sigma(x_1, x_2) .$$

We recall the (term) rewrite semantics (Baader and Nipkow, 1998) of the CFTG  $G = (N, \Sigma, S, P)$ . Since  $G$  is simple, the actual rewriting strategy is irrelevant. The sentential forms of  $G$  are simply  $\text{SF}(G) = T_{N \cup \Sigma}(X)$ . This is slightly more general than necessary (for the semantics of  $G$ ), but the presence of variables in sentential forms will be useful in the next section because it allows us to treat right-hand sides as sentential forms. In essence in a rewrite step we just select a nonterminal  $A \in N$  and an  $A$ -production  $\rho \in P$ . Then we replace an occurrence of  $A$  in the sentential form by the right-hand side of  $\rho$  using second-order substitution.

**Definition 3.** Let  $\xi, \zeta \in \text{SF}(G)$  be sentential forms. Given an  $A$ -production  $\rho = \ell \rightarrow r$  in  $P$  and an

<sup>8</sup>We separate several right-hand sides with ‘|’.



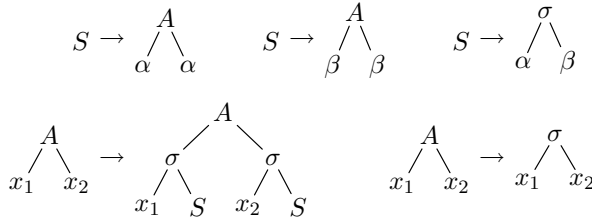


Figure 3: Productions of Example 2.

$A$ -labeled position  $p \in \text{pos}_A(\xi)$  in  $\xi$ , we write  $\xi \Rightarrow_G^{\rho,p} \xi[p \leftarrow r]$ . If there exist  $\rho \in P$  and  $p \in \text{pos}(\xi)$  such that  $\xi \Rightarrow_G^{\rho,p} \zeta$ , then  $\xi \Rightarrow_G \zeta$ .<sup>9</sup> The semantics  $\llbracket G \rrbracket$  of  $G$  is  $\{t \in T_\Sigma \mid S \Rightarrow_G^* t\}$ , where  $\Rightarrow_G^*$  is the reflexive, transitive closure of  $\Rightarrow_G$ .

Two CFTG  $G_1$  and  $G_2$  are (strongly) equivalent if  $\llbracket G_1 \rrbracket = \llbracket G_2 \rrbracket$ . In this contribution we are only concerned with strong equivalence (Chomsky, 1963). Although we recall the string corresponding to a tree later on (via its yield), we will not investigate weak equivalence (Bar-Hillel et al., 1960).

**Example 4.** Reconsider the CFTG  $G_{\text{ex}}$  of Example 2. A derivation to a tree of  $T_\Sigma$  is illustrated in Figure 4. It demonstrates that the final tree in that derivation is in the language  $\llbracket G_{\text{ex}} \rrbracket$  generated by  $G_{\text{ex}}$ .

Finally, let us recall the relation between CFTG and tree adjoining grammars [TAG] (Joshi et al., 1969; Joshi et al., 1975). Joshi et al. (1975) show that TAG are special footed CFTG (Kepser and Rogers, 2011), which are weakly equivalent to monadic CFTG, i.e., CFTG whose nonterminals have rank at most 1 (Mönnich, 1997; Fujiyoshi and Kasai, 2000). Kepser and Rogers (2011) show the strong equivalence of those CFTG to non-strict TAG, which are slightly more powerful than traditional TAG. In general, TAG are a natural formalism to describe the syntax of natural language.<sup>10</sup>

## 4 Normal forms

In this section, we first recall an existing normal form for CFTG. Then we introduce the property of finite ambiguity in the spirit of (Schabes, 1990; Joshi and Schabes, 1992; Kuhlmann and Satta, 2012), which allows us to normalize our CFTG even further. A major tool is a simple production elimination

<sup>9</sup>For all  $k \in \mathbb{N}$  and  $\xi \Rightarrow_G \zeta$  we note that  $\xi \in C_{N \cup \Sigma}(X_k)$  if and only if  $\zeta \in C_{N \cup \Sigma}(X_k)$ .

<sup>10</sup>XTAG Research Group (2001) wrote a TAG for English.

scheme, which we present in detail. From now on, let  $G = (N, \Sigma, S, P)$  be the considered CFTG.

The CFTG  $G$  is *start-separated* if  $\text{pos}_S(r) = \emptyset$  for every production  $\ell \rightarrow r \in P$ . In other words, the start nonterminal  $S$  is not allowed in the right-hand sides of the productions. It is clear that each CFTG can be transformed into an equivalent start-separated CFTG. In such a CFTG we call each production of the form  $S \rightarrow r$  *initial*. From now on, we assume, without loss of generality, that  $G$  is start-separated.

**Example 5.** Let  $G_{\text{ex}} = (N, \Sigma, S, P)$  be the CFTG of Example 2. An equivalent start-separated CFTG is  $G'_{\text{ex}} = (\{S'^{(0)}\} \cup N, \Sigma, S', P \cup \{S' \rightarrow S\})$ .

We start with the growing normal form of Stamer and Otto (2007) and Stamer (2009). It requires that the right-hand side of each non-initial production contains at least two terminal or nonterminal symbols. In particular, it eliminates projection productions  $A(x_1) \rightarrow x_1$  and unit productions, in which the right-hand side has the same shape as the left-hand side (potentially with a different root symbol and a different order of the variables).

**Definition 6.** A production  $\ell \rightarrow r$  is *growing* if  $|\text{pos}_{N \cup \Sigma}(r)| \geq 2$ . The CFTG  $G$  is *growing* if all of its non-initial productions are growing.

The next theorem is Proposition 2 of (Stamer and Otto, 2007). Stamer (2009) provides a full proof.

**Theorem 7.** For every start-separated CFTG there exists an equivalent start-separated, growing CFTG.

**Example 8.** Let us transform the CFTG  $G'_{\text{ex}}$  of Example 5 into growing normal form. We obtain the CFTG  $G''_{\text{ex}} = (\{S'^{(0)}, S^{(0)}, A^{(2)}\}, \Sigma, S', P'')$  where  $P''$  contains  $S' \rightarrow S$  and for each  $\delta \in \{\alpha, \beta\}$

$$S \rightarrow A(\delta, \delta) \mid \sigma(\delta, \delta) \mid \sigma(\alpha, \beta) \quad (1)$$

$$A(x_1, x_2) \rightarrow A(\sigma(x_1, S), \sigma(x_2, S)) \quad (2)$$

$$A(x_1, x_2) \rightarrow \sigma(\sigma(x_1, S), \sigma(x_2, S)) .$$

From now on, we assume that  $G$  is growing. Next, we recall the notion of finite ambiguity from (Schabes, 1990; Joshi and Schabes, 1992; Kuhlmann and Satta, 2012).<sup>11</sup> We distinguish a subset  $\Delta \subseteq \Sigma_0$  of *lexical* symbols, which are the symbols that are preserved by the yield mapping. The yield of a tree is

<sup>11</sup>It should not be confused with the notion of ‘finite ambiguity’ of (Goldstine et al., 1992; Klimann et al., 2004).

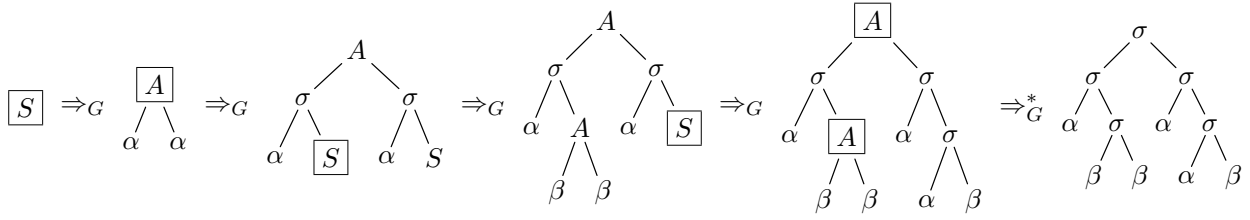


Figure 4: Derivation using the CFTG  $G_{\text{ex}}$  of Example 2. The selected positions are boxed.

a string of lexical symbols. All other symbols are simply dropped (in a pre-order traversal). Formally,  $\text{yd}_\Delta : T_\Sigma \rightarrow \Delta^*$  is such that for all  $t = \sigma(t_1, \dots, t_k)$  with  $\sigma \in \Sigma_k$  and  $t_1, \dots, t_k \in T_\Sigma$

$$\text{yd}_\Delta(t) = \begin{cases} \sigma \text{yd}_\Delta(t_1) \cdots \text{yd}_\Delta(t_k) & \text{if } \sigma \in \Delta \\ \text{yd}_\Delta(t_1) \cdots \text{yd}_\Delta(t_k) & \text{otherwise.} \end{cases}$$

**Definition 9.** The tree language  $L \subseteq T_\Sigma$  has *finite  $\Delta$ -ambiguity* if  $\{t \in L \mid \text{yd}_\Delta(t) = w\}$  is finite for every  $w \in \Delta^*$ .

Roughly speaking, we can say that the set  $L$  has finite  $\Delta$ -ambiguity if each  $w \in \Delta^*$  has finitely many parses in  $L$  (where  $t$  is a parse of  $w$  if  $\text{yd}_\Delta(t) = w$ ). Our example CFTG  $G_{\text{ex}}$  is such that  $\llbracket G_{\text{ex}} \rrbracket$  has finite  $\{\alpha, \beta\}$ -ambiguity (because  $\Sigma_1 = \emptyset$ ).

In this contribution, we want to (strongly) lexicalize CFTG, which means that for each CFTG  $G$  such that  $\llbracket G \rrbracket$  has finite  $\Delta$ -ambiguity, we want to construct an equivalent CFTG such that each non-initial production contains at least one lexical symbol. This is typically called strong lexicalization (Schabes, 1990; Joshi and Schabes, 1992; Kuhlmann and Satta, 2012) because we require strong equivalence.<sup>12</sup> Let us formalize our lexicalization property.

**Definition 10.** The production  $\ell \rightarrow r$  is  *$\Delta$ -lexicalized* if  $\text{pos}_\Delta(r) \neq \emptyset$ . The CFTG  $G$  is  *$\Delta$ -lexicalized* if all its non-initial productions are  $\Delta$ -lexicalized.

Note that the CFTG  $G_{\text{ex}}''$  of Example 8 is not yet  $\{\alpha, \beta\}$ -lexicalized. We will lexicalize it in the next section. To do this in general, we need some auxiliary normal forms. First, we define our simple production elimination scheme, which we will use in the following. Roughly speaking, a non-initial  $A$ -production such that  $A$  does not occur in its right-hand side can be eliminated from  $G$  by applying it in

<sup>12</sup>The corresponding notion for weak equivalence is called weak lexicalization (Joshi and Schabes, 1992).

all possible ways to occurrences in right-hand sides of the remaining productions.

**Definition 11.** Let  $\rho = A(x_1, \dots, x_k) \rightarrow r$  in  $P$  be a non-initial production such that  $\text{pos}_A(r) = \emptyset$ . For every other production  $\rho' = \ell' \rightarrow r'$  in  $P$  and  $J \subseteq \text{pos}_A(r')$ , let  $\rho'_J = \ell' \rightarrow r'[J \leftarrow r]$ . The CFTG  $\text{Elim}(G, \rho) = (N, \Sigma, S, P')$  is such that

$$P' = \bigcup_{\rho' = \ell' \rightarrow r' \in P \setminus \{\rho\}} \{\rho'_J \mid J \subseteq \text{pos}_A(r')\}.$$

In particular,  $\rho'_\emptyset = \rho'$  for every production  $\rho'$ , so every production besides the eliminated production  $\rho$  is preserved. We obtained the CFTG  $G_{\text{ex}}''$  of Example 8 as  $\text{Elim}(G_{\text{ex}}', A(x_1, x_2) \rightarrow \sigma(x_1, x_2))$  from  $G_{\text{ex}}'$  of Example 5.

**Lemma 12.** The CFTG  $G$  and  $G'_\rho = \text{Elim}(G, \rho)$  are equivalent for every non-initial  $A$ -production  $\rho = \ell \rightarrow r$  in  $P$  such that  $\text{pos}_A(r) = \emptyset$ .

*Proof.* Clearly, every single derivation step of  $G'_\rho$  can be simulated by a derivation of  $G$  using potentially several steps. Conversely, a derivation of  $G$  can be simulated directly by  $G'_\rho$  except for derivation steps  $\Rightarrow_G^{\rho, p}$  using the eliminated production  $\rho$ . Since  $S \neq A$ , we know that the nonterminal at position  $p$  was generated by another production  $\rho'$ . In the given derivation of  $G$  we examine which nonterminals in the right-hand side of the instance of  $\rho'$  were replaced using  $\rho$ . Let  $J$  be the set of positions corresponding to those nonterminals (thus  $p \in J$ ). Then instead of applying  $\rho'$  and potentially several times  $\rho$ , we equivalently apply  $\rho'_J$  of  $G'_\rho$ .  $\square$

In the next normalization step we use our production elimination scheme. The goal is to make sure that non-initial monic productions (i.e., productions of which the right-hand side contains at most one nonterminal) contain at least one lexical symbol. We define the relevant property and then present

the construction. A sentential form  $\xi \in \text{SF}(G)$  is *monic* if  $|\text{pos}_N(\xi)| \leq 1$ . The set of all monic sentential forms is denoted by  $\text{SF}_{\leq 1}(G)$ . A production  $\ell \rightarrow r$  is monic if  $r$  is monic. The next construction is similar to the simultaneous removal of epsilon-productions  $A \rightarrow \varepsilon$  and unit productions  $A \rightarrow B$  for context-free grammars (Hopcroft et al., 2001). Instead of computing the closure under those productions, we compute a closure under non- $\Delta$ -lexicalized productions.

**Theorem 13.** If  $\llbracket G \rrbracket$  has finite  $\Delta$ -ambiguity, then there exists an equivalent CFTG such that all its non-initial monic productions are  $\Delta$ -lexicalized.

*Proof.* Without loss of generality, we assume that  $G$  is start-separated and growing by Theorem 7. Moreover, we assume that each nonterminal is useful. For every  $A \in N$  with  $A \neq S$ , we compute all monic sentential forms without a lexical symbol that are reachable from  $A(x_1, \dots, x_k)$ , where  $k = \text{rk}(A)$ . Formally, let

$$\Xi_A = \{ \xi \in \text{SF}_{\leq 1}(G) \mid A(x_1, \dots, x_k) \Rightarrow_{G'}^+ \xi \} ,$$

where  $\Rightarrow_{G'}^+$  is the transitive closure of  $\Rightarrow_{G'}$  and the CFTG  $G' = (N, \Sigma, S, P')$  is such that  $P'$  contains exactly the non- $\Delta$ -lexicalized productions of  $P$ . The set  $\Xi_A$  is finite since only finitely many non- $\Delta$ -lexicalized productions can be used due to the finite  $\Delta$ -ambiguity of  $\llbracket G \rrbracket$ . Moreover, no sentential form in  $\Xi_A$  contains  $A$  for the same reason and the fact that  $G$  is growing. We construct the CFTG  $G_1 = (N, \Sigma, S, P \cup P_1)$  such that

$$P_1 = \{ A(x_1, \dots, x_k) \rightarrow \xi \mid A \in N_k, \xi \in \Xi_A \} .$$

Clearly,  $G$  and  $G_1$  are equivalent. Next, we eliminate all productions of  $P_1$  from  $G_1$  using Lemma 12 to obtain an equivalent CFTG  $G_2$  with the productions  $P_2$ . In the final step, we drop all non- $\Delta$ -lexicalized monic productions of  $P_2$  to obtain the CFTG  $\overline{G}$ , in which all monic productions are  $\Delta$ -lexicalized. It is easy to see that  $\overline{G}$  is growing, start-separated, and equivalent to  $G_2$ .  $\square$

The CFTG  $G''_{\text{ex}}$  only has  $\{\alpha, \beta\}$ -lexicalized non-initial monic productions, so we use a new example.

**Example 14.** Let  $(\{S^{(0)}, A^{(1)}, B^{(1)}\}, \Sigma, S, P)$  be the CFTG such that  $\Sigma = \{\sigma^{(2)}, \alpha^{(0)}, \beta^{(0)}\}$  and

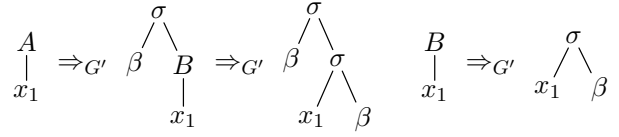


Figure 5: The relevant derivations using only productions that are not  $\Delta$ -lexicalized (see Example 14).

$P$  contains the productions

$$\begin{aligned} A(x_1) &\rightarrow \sigma(\beta, B(x_1)) & B(x_1) &\rightarrow \sigma(x_1, \beta) & (3) \\ B(x_1) &\rightarrow \sigma(\alpha, A(x_1)) & S &\rightarrow A(\alpha) . \end{aligned}$$

This CFTG  $G_{\text{ex}2}$  is start-separated and growing. Moreover, all its productions are monic, and  $\llbracket G_{\text{ex}2} \rrbracket$  is finitely  $\Delta$ -ambiguous for the set  $\Delta = \{\alpha\}$  of lexical symbols. Then the productions (3) are non-initial and not  $\Delta$ -lexicalized. So we can run the construction in the proof of Theorem 13. The relevant derivations using only non- $\Delta$ -lexicalized productions are shown in Figure 5. We observe that  $|\Xi_A| = 2$  and  $|\Xi_B| = 1$ , so we obtain the CFTG  $(\{S^{(0)}, B^{(1)}\}, \Sigma, S, P')$ , where  $P'$  contains<sup>13</sup>

$$\begin{aligned} S &\rightarrow \sigma(\beta, B(\alpha)) \mid \sigma(\beta, \sigma(\alpha, \beta)) \\ B(x_1) &\rightarrow \sigma(\alpha, \sigma(\beta, B(x_1))) \\ B(x_1) &\rightarrow \sigma(\alpha, \sigma(\beta, \sigma(x_1, \beta))) . \end{aligned} \quad (4)$$

We now do one more normalization step before we present our lexicalization. We call a production  $\ell \rightarrow r$  *terminal* if  $r \in T_\Sigma(X)$ ; i.e., it does not contain nonterminal symbols. Next, we show that for each CFTG  $G$  such that  $\llbracket G \rrbracket$  has finite  $\Delta$ -ambiguity we can require that each non-initial terminal production contains at least two occurrences of  $\Delta$ -symbols.

**Theorem 15.** If  $\llbracket G \rrbracket$  has finite  $\Delta$ -ambiguity, then there exists an equivalent CFTG  $(N, \Sigma, S, P')$  such that  $|\text{pos}_\Delta(r)| \geq 2$  for all its non-initial terminal productions  $\ell \rightarrow r \in P'$ .

*Proof.* Without loss of generality, we assume that  $G$  is start-separated and growing by Theorem 7. Moreover, we assume that each nonterminal is useful and that each of its non-initial monic productions is  $\Delta$ -lexicalized by Theorem 13. We obtain the desired CFTG by simply eliminating each non-initial terminal production  $\ell \rightarrow r \in P$  such that  $|\text{pos}_\Delta(r)| = 1$ . By Lemma 12 the obtained CFTG

<sup>13</sup>The nonterminal  $A$  became useless, so we just removed it.

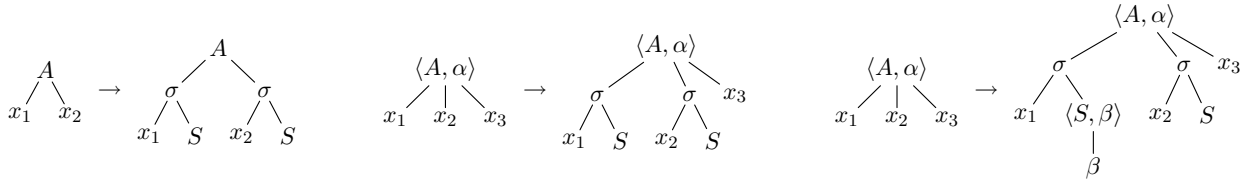


Figure 6: Production  $\rho = \ell \rightarrow r$  of (2) [left], a corresponding production  $\rho_\alpha$  of  $P'$  [middle] with right-hand side  $r_{\alpha,2}$ , and a corresponding production of  $P'''$  [right] with right-hand side  $(\bar{r}_{\alpha,2})_\beta$  (see Theorem 17).

is equivalent to  $G$ . The elimination process terminates because a new terminal production can only be constructed from a monic production and a terminal production or several terminal productions, but those combinations already contain two occurrences of  $\Delta$ -symbols since non-initial monic productions are already  $\Delta$ -lexicalized.  $\square$

**Example 16.** Reconsider the CFTG obtained in Example 14. Recall that  $\Delta = \{\alpha\}$ . Production (4) is the only non-initial terminal production that violates the requirement of Theorem 15. We eliminate it and obtain the CFTG with the productions

$$\begin{aligned} S &\rightarrow \sigma(\beta, B(\alpha)) \mid \sigma(\beta, \sigma(\alpha, \beta)) \\ S &\rightarrow \sigma(\beta, \sigma(\alpha, \sigma(\beta, \sigma(\alpha, \beta)))) \\ B(x_1) &\rightarrow \sigma(\alpha, \sigma(\beta, B(x_1))) \\ B(x_1) &\rightarrow \sigma(\alpha, \sigma(\beta, \sigma(\alpha, \sigma(\beta, \sigma(x_1, \beta))))). \end{aligned}$$

## 5 Lexicalization

In this section, we present the main lexicalization step, which lexicalizes non-monic productions. We assume that  $\llbracket G \rrbracket$  has finite  $\Delta$ -ambiguity and is normalized according to the results of Section 4: no useless nonterminals, start-separated, growing (see Theorem 7), non-initial monic productions are  $\Delta$ -lexicalized (see Theorem 13), and non-initial terminal productions contain at least two occurrences of  $\Delta$ -symbols (see Theorem 15).

The basic idea of the construction is that we guess a lexical symbol for each non- $\Delta$ -lexicalized production. The guessed symbol is put into a new parameter of a nonterminal. It will be kept in the parameter until we reach a terminal production, where we exchange the same lexical symbol by the parameter. This is the reason why we made sure that we have two occurrences of lexical symbols in the terminal productions. After we exchanged one for a parameter, the resulting terminal production is

still  $\Delta$ -lexicalized. Lexical items that are guessed for distinct (occurrences of) productions are transported to distinct (occurrences of) terminal productions [cf. Section 3 of (Potthoff and Thomas, 1993) and page 346 of (Hoogbeem and ten Pas, 1997)].

**Theorem 17.** For every CFTG  $G$  such that  $\llbracket G \rrbracket$  has finite  $\Delta$ -ambiguity there exists an equivalent  $\Delta$ -lexicalized CFTG.

*Proof.* We can assume that  $G = (N, \Sigma, S, P)$  has the properties mentioned before the theorem without loss of generality. We let  $N' = N \times \Delta$  be a new set of nonterminals such that  $\text{rk}(\langle A, \delta \rangle) = \text{rk}(A) + 1$  for every  $A \in N$  and  $\delta \in \Delta$ . Intuitively,  $\langle A, \delta \rangle$  represents the nonterminal  $A$ , which has the lexical symbol  $\delta$  in its last (new) parameter. This parameter is handed to the (lexicographically) first nonterminal in the right-hand side until it is resolved in a terminal production. Formally, for each right-hand side  $r \in T_{N \cup N' \cup \Sigma}(X)$  such that  $\text{pos}_N(r) \neq \emptyset$  (i.e., it contains an original nonterminal), each  $k \in \mathbb{N}$ , and each  $\delta \in \Delta$ , let  $r_{\delta,k}$  and  $\bar{r}_\delta$  be such that

$$\begin{aligned} r_{\delta,k} &= r[\langle B, \delta \rangle(r_1, \dots, r_n, x_{k+1})]_p \\ \bar{r}_\delta &= r[\langle B, \delta \rangle(r_1, \dots, r_n, \delta)]_p, \end{aligned}$$

where  $p$  is the lexicographically smallest element of  $\text{pos}_N(r)$  and  $r|_p = B(r_1, \dots, r_n)$  with  $B \in N$  and  $r_1, \dots, r_n \in T_{N \cup N' \cup \Sigma}(X)$ . For each nonterminal  $A$ -production  $\rho = \ell \rightarrow r$  in  $P$  let

$$\rho_\delta = \langle A, \delta \rangle(x_1, \dots, x_{k+1}) \rightarrow r_{\delta,k},$$

where  $k = \text{rk}(A)$ . This construction is illustrated in Figure 6. Roughly speaking, we select the lexicographically smallest occurrence of a nonterminal in the right-hand side and pass the lexical symbol  $\delta$  in the extra parameter to it. The extra parameter is used in terminal productions, so let  $\rho = \ell \rightarrow r$  in  $P$

$$S \rightarrow \begin{array}{c} \sigma \\ \alpha \quad \alpha \end{array} \quad \langle S, \alpha \rangle \begin{array}{c} | \\ x_1 \end{array} \rightarrow \begin{array}{c} \sigma \\ x_1 \quad \alpha \end{array}$$

Figure 7: Original terminal production  $\rho$  from (1) [left] and the production  $\bar{\rho}$  (see Theorem 17).

be a terminal  $A$ -production. Then we define

$$\bar{\rho} = \langle A, r(p) \rangle (x_1, \dots, x_{k+1}) \rightarrow r[x_{k+1}]_p ,$$

where  $p$  is the lexicographically smallest element of  $\text{pos}_\Delta(r)$  and  $k = \text{rk}(A)$ . This construction is illustrated in Figure 7. With these productions we obtain the CFTG  $G' = (N \cup N', \Sigma, S, \bar{P})$ , where  $\bar{P} = P \cup P' \cup P''$  and

$$P' = \bigcup_{\substack{\rho=\ell \rightarrow r \in P \\ \ell \neq S, \text{pos}_N(r) \neq \emptyset}} \{\rho_\delta \mid \delta \in \Delta\} \quad P'' = \bigcup_{\substack{\rho=\ell \rightarrow r \in P \\ \ell \neq S, \text{pos}_N(r) = \emptyset}} \{\bar{\rho}\} .$$

It is easy to prove that those new productions manage the desired transport of the extra parameter if it holds the value indicated in the nonterminal.

Finally, we replace each non-initial non- $\Delta$ -lexicalized production in  $G'$  by new productions that guess a lexical symbol and add it to the new parameter of the (lexicographically) first nonterminal of  $N$  in the right-hand side. Formally, we let

$$\begin{aligned} \bar{P}_{\text{nil}} &= \{\ell \rightarrow r \in \bar{P} \mid \ell \neq S, \text{pos}_\Delta(r) = \emptyset\} \\ P''' &= \{\ell \rightarrow \bar{r}_\delta \mid \ell \rightarrow r \in \bar{P}_{\text{nil}}, \delta \in \Delta\} , \end{aligned}$$

of which  $P'''$  is added to the productions. Note that each production  $\ell \rightarrow r \in \bar{P}_{\text{nil}}$  contains at least one occurrence of a nonterminal of  $N$  (because all monic productions of  $G$  are  $\Delta$ -lexicalized). Now all non-initial non- $\Delta$ -lexicalized productions from  $\bar{P}$  can be removed, so we obtain the CFTG  $G''$ , which is given by  $(N \cup N', \Sigma, S, R)$  with  $R = (\bar{P} \cup P''') \setminus \bar{P}_{\text{nil}}$ . It can be verified that  $G''$  is  $\Delta$ -lexicalized and equivalent to  $G$  (using the provided argumentation).  $\square$

Instead of taking the lexicographically smallest element of  $\text{pos}_N(r)$  or  $\text{pos}_\Delta(r)$  in the previous proof, we can take any fixed element of that set. In the definition of  $P'$  we can change  $\text{pos}_N(r) \neq \emptyset$  to  $|\text{pos}_\Delta(r)| \leq 1$ , and simultaneously in the definition of  $P''$  change  $\text{pos}_N(r) = \emptyset$  to  $|\text{pos}_\Delta(r)| \geq 2$ . With the latter changes the guessed lexical item is only transported until it is resolved in a production with at least two lexical items.

**Example 18.** For the last time, we consider the CFTG  $G''_{\text{ex}}$  of Example 8. We already illustrated the parts of the construction of Theorem 17 in Figures 6 and 7. The obtained  $\{\alpha, \beta\}$ -lexicalized CFTG has the following 25 productions for all  $\delta, \delta' \in \{\alpha, \beta\}$ :

$$\begin{aligned} S' &\rightarrow S \\ S &\rightarrow A(\delta, \delta) \mid \sigma(\delta, \delta) \mid \sigma(\alpha, \beta) \\ S_\delta(x_1) &\rightarrow A_\delta(\delta', \delta', x_1) \mid \sigma(x_1, \delta) \\ S_\alpha(x_1) &\rightarrow \sigma(x_1, \beta) \\ A(x_1, x_2) &\rightarrow A_\delta(\sigma(x_1, S), \sigma(x_2, S), \delta) \\ A_\delta(x_1, x_2, x_3) &\rightarrow A_\delta(\sigma(x_1, S_\delta(\delta')), \sigma(x_2, S), x_3) \\ A(x_1, x_2) &\rightarrow \sigma(\sigma(x_1, S_\delta(\delta)), \sigma(x_2, S)) \\ A_\delta(x_1, x_2, x_3) &\rightarrow \sigma(\sigma(x_1, S_\delta(x_3)), \sigma(x_2, S_\delta(\delta'))) , \end{aligned} \quad (5)$$

where  $A_\delta = \langle A, \delta \rangle$  and  $S_\delta = \langle S, \delta \rangle$ .

If we change the lexicalization construction as indicated before this example, then all the productions  $S_\delta(x_1) \rightarrow A_\delta(\delta', \delta', x_1)$  are replaced by the productions  $S_\delta(x_1) \rightarrow A(x_1, \delta)$ . Moreover, the productions (5) can be replaced by the productions  $A(x_1, x_2) \rightarrow A(\sigma(x_1, S_\delta(\delta)), \sigma(x_2, S))$ , and then the nonterminals  $A_\delta$  and their productions can be removed, which leaves only 15 productions.

## Conclusion

For  $k \in \mathbb{N}$ , let  $\text{CFTG}(k)$  be the set of those CFTG whose nonterminals have rank at most  $k$ . Since the normal form constructions preserve the nonterminal rank, the proof of Theorem 17 shows that  $\text{CFTG}(k)$  are strongly lexicalized by  $\text{CFTG}(k+1)$ . Kepser and Rogers (2011) show that non-strict TAG are strongly equivalent to  $\text{CFTG}(1)$ . Hence, non-strict TAG are strongly lexicalized by  $\text{CFTG}(2)$ .

It follows from Section 6 of Engelfriet et al. (1980) that the classes  $\text{CFTG}(k)$  with  $k \in \mathbb{N}$  induce an infinite hierarchy of string languages, but it remains an open problem whether the rank increase in our lexicalization construction is necessary.

Gómez-Rodríguez et al. (2010) show that well-nested LCFRS of maximal fan-out  $k$  can be parsed in time  $O(n^{2k+2})$ , where  $n$  is the length of the input string  $w \in \Delta^*$ . From this result we conclude that  $\text{CFTG}(k)$  can be parsed in time  $O(n^{2k+4})$ , in the sense that we can produce a parse tree  $t$  that is generated by the CFTG with  $\text{yd}_\Delta(t) = w$ . It is not clear yet whether lexicalized  $\text{CFTG}(k)$  can be parsed more efficiently in practice.

## References

- Franz Baader and Tobias Nipkow. 1998. *Term Rewriting and All That*. Cambridge University Press.
- Yehoshua Bar-Hillel, Haim Gaifman, and Eli Shamir. 1960. On categorial and phrase-structure grammars. *Bulletin of the Research Council of Israel*, 9F(1):1–16.
- Norbert Blum and Robert Koch. 1999. Greibach normal form transformation revisited. *Inform. and Comput.*, 150(1):112–118.
- John Chen. 2001. *Towards Efficient Statistical Parsing using Lexicalized Grammatical Information*. Ph.D. thesis, University of Delaware, Newark, USA.
- Noam Chomsky. 1963. Formal properties of grammar. In R. Duncan Luce, Robert R. Bush, and Eugene Galanter, editors, *Handbook of Mathematical Psychology*, volume 2, pages 323–418. John Wiley and Sons, Inc.
- Joost Engelfriet, Grzegorz Rozenberg, and Giora Slutzki. 1980. Tree transducers, L systems, and two-way machines. *J. Comput. System Sci.*, 20(2):150–202.
- Michael J. Fischer. 1968. Grammars with macro-like productions. In *Proc. 9th Ann. Symp. Switching and Automata Theory*, pages 131–142. IEEE Computer Society.
- Akio Fujiyoshi. 2005. Epsilon-free grammars and lexicalized grammars that generate the class of the mildly context-sensitive languages. In *Proc. 7th Int. Workshop Tree Adjoining Grammar and Related Formalisms*, pages 16–23.
- Akio Fujiyoshi and Takumi Kasai. 2000. Spinal-formed context-free tree grammars. *Theory Comput. Syst.*, 33(1):59–83.
- Ferenc Gécseg and Magnus Steinby. 1984. *Tree Automata*. Akadémiai Kiadó, Budapest.
- Ferenc Gécseg and Magnus Steinby. 1997. Tree languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, volume 3, chapter 1, pages 1–68. Springer.
- Jonathan Goldstine, Hing Leung, and Detlef Wotschke. 1992. On the relation between ambiguity and nondeterminism in finite automata. *Inform. and Comput.*, 100(2):261–270.
- Carlos Gómez-Rodríguez, Marco Kuhlmann, and Giorgio Satta. 2010. Efficient parsing of well-nested linear context-free rewriting systems. In *Proc. Ann. Conf. North American Chapter of the ACL*, pages 276–284. Association for Computational Linguistics.
- Hendrik Jan Hoogeboom and Paulien ten Pas. 1997. Monadic second-order definable text languages. *Theory Comput. Syst.*, 30(4):335–354.
- John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. 2001. *Introduction to automata theory, languages, and computation*. Addison-Wesley series in computer science. Addison Wesley, 2nd edition.
- Aravind K. Joshi, S. Rao Kosaraju, and H. Yamada. 1969. String adjunct grammars. In *Proc. 10th Ann. Symp. Switching and Automata Theory*, pages 245–262. IEEE Computer Society.
- Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *J. Comput. System Sci.*, 10(1):136–163.
- Aravind K. Joshi and Yves Schabes. 1992. Tree-adjointing grammars and lexicalized grammars. In Maurice Nivat and Andreas Podelski, editors, *Tree Automata and Languages*. North-Holland.
- Aravind K. Joshi and Yves Schabes. 1997. Tree-adjointing grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Beyond Words*, volume 3 of *Handbook of Formal Languages*, pages 69–123. Springer.
- Makoto Kanazawa. 2009. The convergence of well-nested mildly context-sensitive grammar formalisms. Invited talk at the 14th Int. Conf. Formal Grammar. slides available at: [research.nii.ac.jp/~kanazawa](http://research.nii.ac.jp/~kanazawa).
- Makoto Kanazawa and Ryo Yoshinaka. 2005. Lexicalization of second-order ACGs. Technical Report NII-2005-012E, National Institute of Informatics, Tokyo, Japan.
- Stephan Kepser and James Rogers. 2011. The equivalence of tree adjoining grammars and monadic linear context-free tree grammars. *J. Log. Lang. Inf.*, 20(3):361–384.
- Ines Klimann, Sylvain Lombardy, Jean Mairesse, and Christophe Prieur. 2004. Deciding unambiguity and sequentiality from a finitely ambiguous max-plus automaton. *Theoret. Comput. Sci.*, 327(3):349–373.
- Marco Kuhlmann. 2010. *Dependency Structures and Lexicalized Grammars: An Algebraic Approach*, volume 6270 of LNAI. Springer.
- Marco Kuhlmann and Mathias Mohl. 2006. Extended cross-serial dependencies in tree adjoining grammars. In *Proc. 8th Int. Workshop Tree Adjoining Grammars and Related Formalisms*, pages 121–126. ACL.
- Marco Kuhlmann and Giorgio Satta. 2012. Tree-adjointing grammars are not closed under strong lexicalization. *Comput. Linguist.* available at: [dx.doi.org/10.1162/COLI\\_a\\_00090](http://dx.doi.org/10.1162/COLI_a_00090).
- Uwe Mönnich. 1997. Adjunction as substitution: An algebraic formulation of regular, context-free and tree adjoining languages. In *Proc. 3rd Int. Conf. Formal Grammar*, pages 169–178. Université de Provence, France. available at: [arxiv.org/abs/cmp-lg/9707012v1](http://arxiv.org/abs/cmp-lg/9707012v1).
- Uwe Mönnich. 2010. Well-nested tree languages and attributed tree transducers. In *Proc. 10th Int. Conf. Tree Adjoining Grammars and Related Formalisms*. Yale University. available at: [www2.research.att.com/~srini/TAG+10/papers/uwe.pdf](http://www2.research.att.com/~srini/TAG+10/papers/uwe.pdf).

- Andreas Potthoff and Wolfgang Thomas. 1993. Regular tree languages without unary symbols are star-free. In *Proc. 9th Int. Symp. Fundamentals of Computation Theory*, volume 710 of LNCS, pages 396–405. Springer.
- William C. Rounds. 1969. Context-free grammars on trees. In *Proc. 1st ACM Symp. Theory of Comput.*, pages 143–148. ACM.
- William C. Rounds. 1970. Tree-oriented proofs of some theorems on context-free and indexed languages. In *Proc. 2nd ACM Symp. Theory of Comput.*, pages 109–116. ACM.
- Yves Schabes. 1990. *Mathematical and Computational Aspects of Lexicalized Grammars*. Ph.D. thesis, University of Pennsylvania, Philadelphia, USA.
- Yves Schabes, Anne Abeillé, and Aravind K. Joshi. 1988. Parsing strategies with ‘lexicalized’ grammars: Application to tree adjoining grammars. In *Proc. 12th Int. Conf. Computational Linguistics*, pages 578–583. John von Neumann Society for Computing Sciences, Budapest.
- Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar: A cubic-time parsable formalism that lexicalizes context-free grammars without changing the trees produced. *Comput. Linguist.*, 21(4):479–513.
- Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoret. Comput. Sci.*, 88(2):191–229.
- Heiko Stamer. 2009. *Restarting Tree Automata: Formal Properties and Possible Variations*. Ph.D. thesis, University of Kassel, Germany.
- Heiko Stamer and Friedrich Otto. 2007. Restarting tree automata and linear context-free tree languages. In *Proc. 2nd Int. Conf. Algebraic Informatics*, volume 4728 of LNCS, pages 275–289. Springer.
- K. Vijay-Shanker, David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *Proc. 25th Ann. Meeting of the Association for Computational Linguistics*, pages 104–111. Association for Computational Linguistics.
- XTAG Research Group. 2001. A lexicalized tree adjoining grammar for English. Technical Report IRCS-01-03, University of Pennsylvania, Philadelphia, USA.
- Ryo Yoshinaka. 2006. *Extensions and Restrictions of Abstract Categorical Grammars*. Ph.D. thesis, University of Tokyo.

# Tweet Recommendation with Graph Co-Ranking

Rui Yan<sup>†</sup>

<sup>†</sup>Department of Computer Science and Technology, Peking University, Beijing 100871, China  
r.yan@pku.edu.cn

Mirella Lapata<sup>‡</sup>

<sup>‡</sup>Institute for Language, Cognition and Computation, University of Edinburgh, Edinburgh EH8 9AB, UK  
mlap@inf.ed.ac.uk

Xiaoming Li<sup>†,‡</sup>

<sup>‡</sup>State Key Laboratory of Software Development Environment, Beihang University, Beijing 100083, China  
lxm@pku.edu.cn

## Abstract

As one of the most popular micro-blogging services, Twitter attracts millions of users, producing millions of tweets daily. Shared information through this service spreads faster than would have been possible with traditional sources, however the proliferation of user-generated content poses challenges to browsing and finding valuable information. In this paper we propose a graph-theoretic model for tweet recommendation that presents users with items they may have an interest in. Our model ranks tweets and their authors *simultaneously* using several networks: the social network connecting the users, the network connecting the tweets, and a third network that ties the two together. Tweet and author entities are ranked following a co-ranking algorithm based on the intuition that there is a mutually reinforcing relationship between tweets and their authors that could be reflected in the rankings. We show that this framework can be parametrized to take into account user preferences, the popularity of tweets and their authors, and diversity. Experimental evaluation on a large dataset shows that our model outperforms competitive approaches by a large margin.

## 1 Introduction

Online micro-blogging services have revolutionized the way people discover, share, and distribute information. *Twitter* is perhaps the most popular such service with over 140 million active users as of

2012.<sup>1</sup> Twitter enables users to send and read text-based posts of up to 140 characters, known as *tweets*. Twitter users follow others or are followed. Being a follower on Twitter means that the user receives all the tweets from those she follows. Common practice of responding to a tweet has evolved into a well-defined markup culture (e.g., RT stands for retweet, '@' followed by an identifier indicates the user). The strict limit of 140 characters allows for quick and immediate communication in real time, whilst enforcing brevity. Moreover, the retweet mechanism empowers users to spread information of their choice beyond the reach of their original followers.

Twitter has become a prominent broadcasting medium, taking priority over traditional news sources (Teevan et al., 2011). Shared information through this channel spreads faster than would have been possible with conventional news sites or RSS feeds and can reach a far wider population base. However, the proliferation of user-generated content comes at a price. Over 340 millions of tweets are being generated daily amounting to thousands of tweets per second!<sup>2</sup> Twitter's own search engine handles more than 1.6 billion search queries per day.<sup>3</sup> This enormous amount of data renders it infeasible to browse the entire Twitter network; even if this was possible, it would be extremely difficult for users to find information they are interested in. A hypothetical tweet recommendation system could

<sup>1</sup>For details see <http://blog.twitter.com/2012/03/twitter-turns-six.html>

<sup>2</sup>In fact, the peak record is 6,939 tweets per second, reported by <http://blog.twitter.com/2011/03/numbers.html>.

<sup>3</sup>See <http://engineering.twitter.com/2011/05/engineering-behind-twitters-new-search.html>



alleviate this acute information overload, e.g., by limiting the stream of tweets to those of interest to the user, or by discovering intriguing content outside the user’s following network.

The tweet recommendation task is challenging for several reasons. Firstly, Twitter does not merely consist of a set of tweets. Rather, it contains many latent networks including the *following* relationships among users and the *retweeting* linkage (which indicates information diffusion). Secondly, the recommendations ought to be of interest to the user and likely to attract user response (e.g., to be retweeted). Thirdly, recommendations should be personalized (Cho and Schonfeld, 2007; Yan et al., 2011), avoid redundancy, and demonstrate diversity. In this paper we present a graph-theoretic approach to tweet recommendation that attempts to address these challenges.

Our recommender operates over a heterogeneous network that connects the users (or authors) and the tweets they produce. The user network represents links among authors based on their *following* behavior, whereas the tweet network connects tweets based on content similarity. A third bipartite graph ties the two together. Tweet and author entities in this network are ranked simultaneously following a co-ranking algorithm (Zhou et al., 2007). The main intuition behind co-ranking is that there is a mutually reinforcing relationship between authors and tweets that could be reflected in the rankings. Tweets are important if they are related to other important tweets and authored by important users who in turn are related to other important users. The model exploits this mutually reinforcing relationship between tweets and their authors and couples two random walks, one on the tweet graph and one on the author graph, into a combined one. Rather than creating a global ranking over all tweets in a collection, we extend this framework to individual users and produce personalized recommendations. Moreover, we incorporate diversity by allowing the random walk on the tweet graph to be time-variant (Mei et al., 2010).

Experimental results on a real-world dataset consisting of 364,287,744 tweets from 9,449,542 users show that the co-ranking approach substantially improves performance over the state of the art. We obtain a relative improvement of 18.3% (in nDCG) and 7.8% (in MAP) over the best comparison system.

## 2 Related Work

**Tweet Search** Given the large amount of tweets being posted daily, ranking strategies have become extremely important for retrieving information quickly. Many websites currently offer a real-time search service which returns ranked lists of Twitter posts or shared links according to user queries. Ranking methods used by these sites employ three criteria, namely recency, popularity and content relevance (Dong et al., 2010). State-of-art tweet retrieval methods include a linear regression model biased towards text quality with a regularization factor inspired by the hypothesis that documents similar in content may have similar quality (Huang et al., 2011). Duan et al. (2010) learn a ranking model using SVMs and features based on tweet content, the relations among users, and tweet specific characteristics (e.g., urls, number of retweets).

**Tweet Recommendation** Previous work has also focused on tweet recommendation systems, assuming no explicit query is provided by the users. Collaborative filtering is perhaps the most obvious method for recommending tweets (Hannon et al., 2010). Chen et al. (2010) investigate how to select interesting URLs linked from Twitter and recommend the top ranked ones to users. Their recommender takes three dimensions into account: the source, the content topic, and social voting. Similarly, Abel et al. (2011a; 2011b; 2011c) recommend external websites linked to Twitter. Their method incorporates user profile modeling and temporal recency, but they do not utilize the social networks among users. R. et al. (2009) propose a diffusion-based recommendation framework especially for tweets representing critical events by constructing a diffusion graph. Hong et al. (2011) recommend tweets based on popularity related features. Ramage et al. (2010) investigate which topics users are interested in following a Labeled-LDA approach, by deciding whether a user is in the followee list of a given user or not. Uysal and Croft (2011) estimate the likelihood of a tweet being reposted from a user-centric perspective.

Our work also develops a tweet recommendation system. Our model exploits the information provided by the tweets and the underlying social networks in a unified co-ranking framework. Although

these sources have been previously used to search or recommend tweets, our model considers them simultaneously and produces a ranking that is informed by both. Furthermore, we argue that the graph-theoretic framework upon which co-ranking operates is beneficial as it allows to incorporate personalization (we provide user-specific rankings) and diversity (the ranking is optimized so as to avoid redundancy). The co-ranking framework has been initially developed for measuring scientific impact and modeling the relationship between authors and their publications (Zhou et al., 2007). However, the adaptation of this framework to the tweet recommendation task is novel to our knowledge.

### 3 Tweet Recommendation Framework

Our method operates over a heterogeneous network that connects three graphs representing the tweets, their authors and the relationships between them. Let  $G$  denote the heterogeneous graph with nodes  $V$  and edges  $E$ , and  $G = (V, E) = (V_M \cup V_U, E_M \cup E_U \cup E_{MU})$ .  $G$  is divided into three subgraphs,  $G_M$ ,  $G_U$  and  $G_{MU}$ .  $G_M = (V_M, E_M)$  is a weighted undirected graph representing the tweets and their relationships. Let  $V_M = \{m_i | m_i \in V_M\}$  denote a collection of  $|V_M|$  tweets and  $E_M$  the set of links representing relationships between them. The latter are established by measuring how semantically similar any two tweets are (see Section 3.4 for details).  $G_U = (V_U, E_U)$  is an unweighted directed graph representing the social ties among Twitter users.  $V_U = \{u_i | u_i \in V_U\}$  is the set of users with size  $|V_U|$ . Links  $E_U$  among users are established by observing their *following* behavior.  $G_{MU} = (V_{MU}, E_{MU})$  is an unweighted bipartite graph that ties  $G_M$  and  $G_U$  together and represents tweet-author relationships. The graph consists of nodes  $V_{MU} = V_M \cup V_U$  and edges  $E_{MU}$  connecting each tweet with all of its authors. Typically, a tweet  $m$  is written by only one author  $u$ . However, because of *retweeting* we treat all users involved in reposting a tweet as “co-authors”. The three subnetworks are illustrated in Figure 1.

The framework includes three random walks, one on  $G_M$ , one on  $G_U$  and one on  $G_{MU}$ . A random walk on a graph is a Markov chain, its states being the vertices of the graph. It can be described by a square  $n \times n$  matrix  $\mathbf{M}$ , where  $n$  is the number of vertices in the graph.  $\mathbf{M}$  is a stochastic matrix prescribing

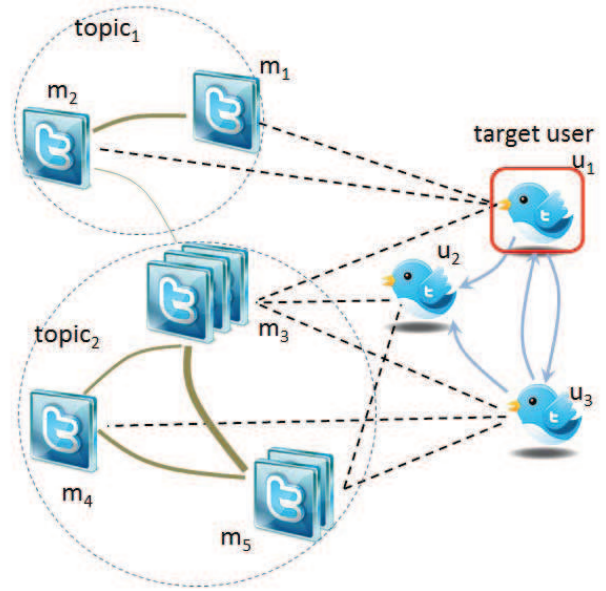


Figure 1: Tweet recommendation based on a co-ranking framework including three sub-networks. The undirected links between tweets indicate semantic correlation. The directed links between users denotes *following*. A bipartite graph (whose edges are shown with dashed lines) ties the tweet and author networks together.

the transition probabilities from one vertex to the next. The framework couples the two random walks on  $G_M$ , and  $G_U$  that rank tweets and their authors in isolation. and allows to obtain a more global ranking by taking into account their mutual dependence. In the following sections we first describe how we obtain the rankings on  $G_M$  and  $G_U$ , and then move on to discuss how the two are coupled.

#### 3.1 Ranking the Tweet Graph

**Popularity** We rank the tweet network following the PageRank paradigm (Brin and Page, 1998). Consider a random walk on  $G_M$  and let  $\mathbf{M}$  be the transition matrix (defined in Section 3.4). Fix some damping factor  $\mu$  and say that at each time step with probability  $(1-\mu)$  we stick to random walking and with probability  $\mu$  we do not make a usual random walk step, but instead jump to any vertex, chosen uniformly at random:

$$\mathbf{m} = (1 - \mu)\mathbf{M}^T\mathbf{m} + \frac{\mu}{|V_M|}\mathbf{1}\mathbf{1}^T \quad (1)$$

Here, vector  $\mathbf{m}$  contains the ranking scores for the vertices in  $G_M$ . The fact that there exists a unique so-

lution to (1) follows from the random walk  $\mathbf{M}$  being ergodic ( $\mu > 0$  guarantees irreducibility, because we can jump to any vertex).  $\mathbf{M}^T$  is the transpose of  $\mathbf{M}$ .  $\mathbf{1}$  is the vector of  $|V_M|$  entries, each being equal to one. Let  $\mathbf{m} \in \mathbf{R}^{V_M}$ ,  $\|\mathbf{m}\|_1 = 1$  be the only solution.

**Personalization** The standard PageRank algorithm performs a random walk, starting from any node, then randomly selects a link from that node to follow considering the weighted matrix  $\mathbf{M}$ , or jumps to a random node with equal probability. It produces a global ranking over all tweets in the collection without taking specific users into account. As there are billions of tweets available on Twitter covering many diverse topics, it is reasonable to assume that an average user will only be interested in a small subset (Qiu and Cho, 2006). We operationalize a user’s topic preference as a vector  $\mathbf{t} = [t_1, t_2, \dots, t_n]_{1 \times n}$ , where  $n$  denotes the number of topics, and  $t_i$  represents the degree of preference for topic  $i$ . The vector  $\mathbf{t}$  is normalized such that  $\sum_{i=1}^n t_i = 1$ . Intuitively, such vectors will be different for different users. Note that user preferences can be also defined at the *tweet* (rather than *topic*) level. Although tweets can illustrate user interests more directly, in most cases a user will only respond to a small fraction of tweets. This means that most tweets will not provide any information relating to a user’s interests. The topic preference vector allows to propagate such information (based on whether a tweet has been reposted or not) to other tweets within the same topic cluster.

Given  $n$  topics, we obtain a topic distribution matrix  $\mathbf{D}$  using Latent Dirichlet Allocation (Blei et al., 2003). Let  $D_{ij}$  denote the probability of tweet  $m_i$  to belong to topic  $t_j$ . Consider a user with a topic preference vector  $\mathbf{t}$  and topic distribution matrix  $\mathbf{D}$ . We calculate the response probability  $\mathbf{r}$  for all tweets for this user as:

$$\mathbf{r} = \mathbf{tD}^T \quad (2)$$

where  $\mathbf{r} = [r_1, r_2, \dots, r_{V_M}]_{1 \times |V_M|}$  represents the response probability vector and  $r_i$  the probability for a user to respond to tweet  $m_i$ . We normalize  $\mathbf{r}$  so that  $\sum_{r_i \in \mathbf{r}} r_i = 1$ . Now, given the observed response probability vector  $\mathbf{r} = [r_1, r_2, \dots, r_w]_{1 \times w}$ , where  $w < |V_M|$  for a given user and the topic distribution matrix  $\mathbf{D}$ , our task is estimate the topic preference vector  $\mathbf{t}$ . We do this using maximum-likelihood

estimation. Assuming a user has responded to  $w$  tweets, we approximate  $\mathbf{t}$  so as to maximize the observed response probability. Let  $\mathbf{r}(\mathbf{t}) = \mathbf{tD}^T$ . Assuming all responses are independent, the probability for  $w$  tweets  $r_1, r_2, \dots, r_w$  is then  $\prod_{i=1}^w r_i(\mathbf{t})$  under a given  $\mathbf{t}$ . The value of  $\mathbf{t}$  is chosen when the probability is maximized:

$$\mathbf{t} = \underset{\mathbf{t}}{\operatorname{argmax}} \left( \prod_{i=1}^w r_i(\mathbf{t}) \right) \quad (3)$$

In a simple random walk, it is assumed that all nodes in the matrix  $\mathbf{M}$  are equi-probable before the walk. In contrast, we use the topic preference vector as a prior on  $\mathbf{M}$ . Let  $\operatorname{Diag}(\mathbf{r})$  denote a diagonal matrix whose eigenvalue is vector  $\mathbf{r}$ . Then  $\mathbf{m}$  becomes:

$$\begin{aligned} \mathbf{m} &= (1 - \mu)[\operatorname{Diag}(\mathbf{r})\mathbf{M}]^T \mathbf{m} + \mu \mathbf{r} \\ &= (1 - \mu)[\operatorname{Diag}(\mathbf{tD}^T)\mathbf{M}]^T \mathbf{m} + \mu \mathbf{tD}^T \end{aligned} \quad (4)$$

**Diversity** We would also like our output to be diverse without redundant information. Unfortunately, equation (4) will have the opposite effect, as it assigns high scores to closely connected node communities. A greedy algorithm such as Maximum Marginal Relevance (Carbonell and Goldstein, 1998; Wan et al., 2007; Wan et al., 2010) may achieve diversity by iteratively selecting the most prestigious or popular vertex and then penalizing the vertices “covered” by those that have been already selected. Rather than adopting a greedy vertex selection method, we follow DivRank (Mei et al., 2010) a recently proposed algorithm that balances popularity and diversity in ranking, based on a time-variant random walk. In contrast to PageRank, DivRank assumes that the transition probabilities change over time. Moreover, it is assumed that the transition probability from one state to another is reinforced by the number of previous visits to that state. At each step, the algorithm creates a dynamic transition matrix  $\mathbf{M}^{(z)}$ . After  $z$  iterations, the matrix becomes:

$$\mathbf{M}^{(z)} = (1 - \mu)\mathbf{M}^{(z-1)} \cdot \mathbf{m}^{(z-1)} + \mu \mathbf{tD}^T \quad (5)$$

and hence,  $\mathbf{m}$  can be calculated as:

$$\mathbf{m}^{(z)} = (1 - \mu)[\operatorname{Diag}(\mathbf{tD}^T)\mathbf{M}^{(z)}]^T \mathbf{m} + \mu \mathbf{tD}^T \quad (6)$$

Equation (5) increases the probability for nodes with higher popularity. Nodes with high weights are

likely to “absorb” the weights of their neighbors directly, and the weights of their neighbors’ neighbors indirectly. The process iteratively adjusts the matrix  $\mathbf{M}$  according to  $\mathbf{m}$  and then updates  $\mathbf{m}$  according to the changed  $\mathbf{M}$ . Essentially, the algorithm favors nodes with high popularity and as time goes by there emerges a *rich-gets-richer* effect (Mei et al., 2010).

### 3.2 Ranking the Author Graph

As mentioned earlier, we build a graph of authors (and obtain the affinity  $\mathbf{U}$ ) using the *following* linkage. We rank the author network using PageRank analogously to equation (1). Besides popularity, we also take personalization into account. Intuitively, users are likely to be interested in their friends even if these are relatively unpopular. Therefore, for each author, we include a vector  $\mathbf{p} = [p_1, p_2, \dots, p_{|V_U|}]_{1 \times |V_U|}$  denoting their preference for other authors. The preference factor for author  $u$  toward other authors  $u_i$  is defined as:

$$p_i^u = \frac{\text{\#tweets from } u_i}{\text{\#tweets of } u} \quad (7)$$

which represents the proportion of tweets inherited from user  $u_i$ . A large  $p_i^u$  means that  $u$  is more likely to respond to  $u_i$ ’s tweets.

In theory, we could also apply DivRank on the author graph. However, as the authors are unique, we assume that they are sufficiently distinct and there is no need to promote diversity.

### 3.3 The Co-Ranking Algorithm

So far we have described how we rank the network of tweets  $G_M$  and their authors  $G_U$  independently following the PageRank paradigm. The co-ranking framework includes a random walk on  $G_M$ ,  $G_U$ , and  $G_{MU}$ . The latter is a bipartite graph representing which tweets are authored by which users. The random walks on  $G_M$  and  $G_U$  are *intra-class* random walks, because take place either within the tweets’ or the users’ networks. The third (combined) random walk on  $G_{MU}$  is an *inter-class* random walk. It is sufficient to describe it by a matrix  $\mathbf{MU}_{|V_M| \times |V_U|}$  and a matrix  $\mathbf{UM}_{|V_U| \times |V_M|}$ , since  $G_{MU}$  is bipartite. One intra-class step changes the probability distribution from  $(\mathbf{m}, \mathbf{0})$  to  $(M\mathbf{m}, \mathbf{0})$  or from  $(\mathbf{0}, \mathbf{u})$  to  $(\mathbf{0}, U\mathbf{u})$ , while one inter-class step changes the probability distribution from  $(\mathbf{m}, \mathbf{u})$  to  $(\mathbf{UM}^T \mathbf{u}, \mathbf{MU}^T \mathbf{m})$ . The

design of  $\mathbf{M}$ ,  $\mathbf{U}$ ,  $\mathbf{MU}$  and  $\mathbf{UM}$  is detailed in Section 3.4.

The two intra-class random walks are coupled using the inter-class random walk on the bipartite graph. The coupling is regulated by  $\lambda$ , a parameter quantifying the importance of  $G_{MU}$  versus  $G_M$  and  $G_U$ . In the extreme case, if  $\lambda$  is set to 0, there is no coupling. This amounts to separately ranking tweets and authors by PageRank. In general,  $\lambda$  represents the extent to which the ranking of tweets and their authors depend on each other.

There are two intuitions behind the co-ranking algorithm: (1) a tweet is important if it associates to other important tweets, and is authored by important users and (2) a user is important if they associate to other important users, and they write important tweets. We formulate these intuitions using the following iterative procedure:

**Step 1** Compute tweet saliency scores:

$$\begin{aligned} \mathbf{m}^{(z+1)} &= (1 - \lambda)([\text{Diag}(\mathbf{r})\mathbf{M}^{(z)}]^T)\mathbf{m}^{(z)} + \lambda\mathbf{UM}^T\mathbf{u}^{(z)} \\ \mathbf{m}^{(z+1)} &= \mathbf{m}^{(z+1)} / \|\mathbf{m}^{(z+1)}\| \end{aligned} \quad (8)$$

**Step 2** Compute author saliency scores:

$$\begin{aligned} \mathbf{u}^{(z+1)} &= (1 - \lambda)([\text{Diag}(\mathbf{p})\mathbf{U}]^T)\mathbf{u}^{(z)} + \lambda\mathbf{MU}^T\mathbf{m}^{(z)} \\ \mathbf{u}^{(z+1)} &= \mathbf{u}^{(z+1)} / \|\mathbf{u}^{(z+1)}\| \end{aligned} \quad (9)$$

Here,  $\mathbf{m}^{(z)}$  and  $\mathbf{u}^{(z)}$  are the ranking vectors for tweets and authors for the  $z$ -th iteration. To guarantee convergence,  $\mathbf{m}$  and  $\mathbf{u}$  are normalized after each iteration. Note that the tweet transition matrix  $\mathbf{M}$  is dynamic due to the computation of diversity while the author transition matrix  $\mathbf{U}$  is static. The algorithm typically converges when the difference between the scores computed at two successive iterations for any tweet/author falls below a threshold  $\epsilon$  (set to 0.001 in this study).

### 3.4 Affinity Matrices

The co-ranking framework is controlled by four affinity matrices:  $\mathbf{M}$ ,  $\mathbf{U}$ ,  $\mathbf{MU}$  and  $\mathbf{UM}$ . In this section we explain how these matrices are defined in more detail.

The tweet graph is an undirected weighted graph, where an edge between two tweets  $m_i$  and  $m_j$  represents their cosine similarity. An adjacency matrix  $\mathbf{M}$

describes the tweet graph where each entry corresponds to the weight of a link in the graph:

$$M_{ij} = \frac{\mathcal{F}(m_i, m_j)}{\sum_k \mathcal{F}(m_i, m_k)}, \mathcal{F}(m_i, m_j) = \frac{\vec{m}_i \cdot \vec{m}_j}{\|\vec{m}_i\| \|\vec{m}_j\|} \quad (10)$$

where  $\mathcal{F}(\cdot)$  is the cosine similarity and  $\vec{m}$  is a term vector corresponding to tweet  $m$ . We treat a tweet as a short document and weight each term with *tf.idf* (Salton and Buckley, 1988), where *tf* is the term frequency and *idf* is the inverse document frequency.

The author graph is a directed graph based on the *following* linkage. When  $u_i$  follows  $u_j$ , we add a link from  $u_i$  to  $u_j$ . Let the indicator function  $I(u_i, u_j)$  denote whether  $u_i$  follows  $u_j$ . The adjacency matrix  $\mathbf{U}$  is then defined as:

$$U_{ij} = \frac{I(u_i, u_j)}{\sum_k I(u_i, u_k)}, I(u_i, u_j) = \begin{cases} 1 & \text{if } e_{ij} \in E_U \\ 0 & \text{if } e_{ij} \notin E_U \end{cases} \quad (11)$$

In the bipartite tweet-author graph  $G_{MU}$ , the entry  $E_{MU}(i, j)$  is an indicator function denoting whether tweet  $m_i$  is authored by user  $u_j$ :

$$\mathcal{A}(m_i, u_j) = \begin{cases} 1 & \text{if } e_{ij} \in E_{MU} \\ 0 & \text{if } e_{ij} \notin E_{MU} \end{cases} \quad (12)$$

Through  $E_{MU}$  we define  $\mathbf{MU}$  and  $\mathbf{UM}$ , using the weight matrices  $\mathbf{MU} = [\bar{W}_{ij}]$  and  $\mathbf{UM} = [\hat{W}_{ji}]$ , containing the conditional probabilities of transitioning from  $m_i$  to  $u_j$  and vice versa:

$$\bar{W}_{ij} = \frac{\mathcal{A}(m_i, u_j)}{\sum_k \mathcal{A}(m_i, u_k)}, \hat{W}_{ji} = \frac{\mathcal{A}(m_i, u_j)}{\sum_k \mathcal{A}(m_k, u_j)} \quad (13)$$

## 4 Experimental Setup

**Data** We crawled Twitter data from 23 seed users (who were later invited to manually evaluate the output of our system). In addition, we collected the data of their *followees* and *followers* by traversing the *following* edges, and exploring all newly included users in the same way until no new users were added. This procedure resulted in a relatively large dataset consisting of 9,449,542 users, 364,287,744 tweets, 596,777,491 links, and 55,526,494 retweets. The crawler monitored the data from 3/25/2011 to 5/30/2011. We used approximately one month of this data for training and the rest for testing.

Before building the graphs (i.e., the tweet graph, the author graph, and the tweet-author graph), the dataset was preprocessed as follows. We removed tweets of low linguistic quality and subsequently discarded users without any linkage to the remaining tweets. We measured linguistic quality following the evaluation framework put forward in Pitler et al. (2010). For instance, we measured the out-of-vocabulary word ratio (as a way of gauging spelling errors), entity coherence, fluency, and so on. We further removed stopwords and performed stemming.

**Parameter Settings** We ran LDA with 500 iterations of Gibbs sampling. The number of topics  $n$  was set to 100 which upon inspection seemed generally coherent and meaningful. We set the damping factor  $\mu$  to 0.15 following the standard PageRank paradigm. We opted for more or less generic parameter values as we did not want to tune our framework to the specific dataset at hand. We examined the parameter  $\lambda$  which controls the balance of the tweet-author graph in more detail. We experimented with values ranging from 0 to 0.9, with a step size of 0.1. Small  $\lambda$  values place little emphasis on the tweet graph, whereas larger values rely more heavily on the author graph. Mid-range values take both graphs into account. Overall, we observed better performance with values larger than 0.4. This suggests that both sources of information — the content of the tweets and their authors — are important for the recommendation task. All our experiments used the same  $\lambda$  value which was set to 0.6.

**System Comparison** We compared our approach against three naive baselines and three state-of-the-art systems recently proposed in the literature. All comparison systems were subject to the same filtering and preprocessing procedures as our own algorithm. Our first baseline ranks tweets randomly (Random). Our second baseline ranks tweets according to token length: longer tweets are ranked higher (Length). The third baseline ranks tweets by the number of times they are reposted assuming that more reposting is better (RTnum). We also compared our method against Duan et al. (2010). Their model (RSVM) ranks tweets based on tweet content features and tweet authority features using the RankSVM algorithm (Joachims, 1999). Our fifth comparison system (DTC) was Uysal and Croft

(2011) who use a decision tree classifier to judge how likely it is for a tweet to be reposted by a specific user. This scenario is similar to ours when ranking tweets by retweet likelihood. Finally, we compared against Huang et al. (2011) who use weighted linear combination (WLC) to grade the relevance of a tweet given a query. We implemented their model without any query-related features as in our setting we do not discriminate tweets depending on their relevance to specific queries.

**Evaluation** We evaluated system output in two ways, i.e., automatically and in a user study. Specifically, we assume that if a tweet is retweeted it is relevant and is thus ranked higher over tweets that have not been reposted. We used our algorithm to predict a ranking for the tweets in the test data which we then compared against a goldstandard ranking based on whether a tweet has been retweeted or not. We measured ranking performance using the normalized Discounted Cumulative Gain (nDCG; Järvelin and Kekäläinen (2002)):

$$\text{nDCG}(k, V_U) = \frac{1}{|V_U|} \sum_{u \in V_U} \frac{1}{Z_u} \sum_{i=1}^k \frac{2^{r_i^u} - 1}{\log(1+i)} \quad (14)$$

where  $V_U$  denotes users,  $k$  indicates the top- $k$  positions in a ranked list, and  $Z_u$  is a normalization factor obtained from a perfect ranking for a particular user.  $r_i^u$  is the relevance score (i.e., 1: retweeted, 0: not retweeted) for the  $i$ -th tweet in the ranking list for user  $u$ .

We also evaluated system output in terms of Mean Average Precision (MAP), under the assumption that retweeted tweets are relevant and the rest irrelevant:

$$\text{MAP} = \frac{1}{|V_U|} \sum_{u \in V_U} \frac{1}{N_u} \sum_{i=1}^k P_i^u \times r_i^u \quad (15)$$

where  $N_u$  is the number of reposted tweets for user  $u$ , and  $P_i^u$  is the precision at  $i$ -th position for user  $u$  (Manning et al., 2008).

The automatic evaluation sketched above does not assess the full potential of our recommendation system. For instance, it is possible for the algorithm to recommend tweets to users with no linkage to their publishers. Such tweets may be of potential interest, however our goldstandard data can only provide information for tweets and users with *following* links.

System	nDCG@5	nDCG@10	nDCG@25	nDCG@50	MAP
Random	0.068	0.111	0.153	0.180	0.167
Length	0.275	0.288	0.298	0.335	0.258
RTNum	0.233	0.219	0.225	0.249	0.239
RSVM	0.392	0.400	0.421	0.444	0.558
DTC	0.441	0.468	0.492	0.473	0.603
WLC	0.404	0.421	0.437	0.464	0.592
CoRank	<b>0.519</b>	<b>0.546</b>	<b>0.550</b>	<b>0.585</b>	<b>0.617</b>

Table 1: Evaluation of tweet ranking output produced by our system and comparison baselines against goldstandard data.

System	nDCG@5	nDCG@10	nDCG@25	nDCG@50	MAP
Random	0.081	0.103	0.116	0.107	0.175
Length	0.291	0.307	0.246	0.291	0.264
RTNum	0.258	0.318	0.343	0.346	0.257
RSVM	0.346	0.443	0.384	0.414	0.447
DTC	0.545	0.565	0.579	0.526	0.554
WLC	0.399	0.447	0.460	0.481	0.506
CoRank	<b>0.567</b>	<b>0.644</b>	<b>0.715</b>	<b>0.643</b>	<b>0.628</b>

Table 2: Evaluation of tweet ranking output produced by our system and comparison baselines against judgments elicited by users.

We therefore asked the 23 users whose Twitter data formed the basis of our corpus to judge the tweets ranked by our algorithm and comparison systems. The users were asked to read the systems’ recommendations and decide for every tweet presented to them whether they would retweet it or not, under the assumption that retweeting takes place when users find the tweet interesting.

In both automatic and human-based evaluations we ranked all tweets in the test data. Then for each date and user we selected the top 50 ones. Our nDCG and MAP results are averages over users and dates.

## 5 Results

Our results are summarized in Tables 1 and 2. Table 1 reports results when model performance is evaluated against the gold standard ranking obtained from the Twitter network. In Table 2 model performance is compared against rankings elicited by users.

As can be seen, the Random method performs worst. This is hardly surprising as it recommends tweets without any notion of their importance or user interest. Length performs considerably better than

System	nDCG@5	nDCG@10	nDCG@25	nDCG@50	MAP
PageRank	0.493	0.481	0.509	0.536	0.604
PersRank	0.501	0.542	<b>0.558</b>	0.560	0.611
DivRank	0.487	0.505	0.518	0.523	0.585
CoRank	<b>0.519</b>	<b>0.546</b>	0.550	<b>0.585</b>	<b>0.617</b>

Table 3: Evaluation of individual system components against goldstandard data.

System	nDCG@5	nDCG@10	nDCG@25	nDCG@50	MAP
PageRank	0.557	0.549	0.623	0.559	0.588
PersRank	0.571	0.595	0.655	0.613	0.601
DivRank	0.538	0.591	0.594	0.547	0.589
CoRank	<b>0.637</b>	<b>0.644</b>	<b>0.715</b>	<b>0.643</b>	<b>0.628</b>

Table 4: Evaluation of individual system components against human judgments.

Random. This might be due to the fact that informativeness is related to tweet length. Using merely the number of retweets does not seem to capture the tweet importance as well as Length. This suggests that highly retweeted posts are not necessarily informative. For example, in our data, the most frequently reposted tweet is a commercial advertisement calling for reposting!

The supervised systems (RSVM, DTC, and WLC) greatly improve performance over the naive baselines. These methods employ standard machine learning algorithms (such as SVMs, decision trees and linear regression) on a large feature space. Aside from the learning algorithm, their main difference lies in the selection of the feature space, e.g., the way content is represented and whether authority is taken into account. DTC performs best on most evaluation criteria. However, neither DTC nor RSVM, or WLC take personalization into account. They generate the same recommendation lists for all users. Our co-ranking algorithm models user interest with respect to the content of the tweets and their publishers. Moreover, it attempts to create diverse output and has an explicit mechanism for minimizing redundancy. In all instances, using both DCG and MAP, it outperforms the comparison systems. Interestingly, the performance of CoRank is better when measured against human judgments. This indicates that users are interested in tweets that fall outside the scope of their followers and that recommendation can improve user experience.

We further examined the contribution of the individual components of our system to the tweet recommendation task. Tables 3 and 4 show how the performance of our co-ranking algorithm varies when considering only tweet popularity using the standard PageRank algorithm, personalization (PersRank), and diversity (DivRank). Note that DivRank is only applied to the tweet graph. The PageRank algorithm on its own makes good recommendations, while incorporating personalization improves the performance substantially, which indicates that individual users show preferences to specific topics or other users. Diversity on its own does not seem to make a difference, however it improves performance when combined with personalization. Intuitively, users are more likely to repost tweets from their followees, or tweets closely related to those retweeted previously.

## 6 Conclusions

We presented a co-ranking framework for a tweet recommendation system that takes popularity, personalization and diversity into account. Central to our approach is the representation of tweets and their users in a heterogeneous network and the ability to produce a global ranking that takes both information sources into account. Our model obtains substantial performance gains over competitive approaches on a large real-world dataset (it improves by 18.3% in DCG and 7.8% in MAP over the best baseline). Our experiments suggest that improvements are due to the synergy of the two information sources (i.e., tweets and their authors). The adopted graph-theoretic framework is advantageous in that it allows to produce user-specific recommendations and incorporate diversity in a unified model. Evaluation with actual Twitter users shows that our recommender can indeed identify interesting information that lies outside the the user’s immediate following network. In the future, we plan to extend the co-ranking framework so as to incorporate information credibility and temporal recency.

**Acknowledgments** This work was partially funded by the Natural Science Foundation of China under grant 60933004, and the Open Fund of the State Key Laboratory of Software Development Environment under grant SKLSDE-2010KF-03. Rui Yan was supported by a MediaTek Fellowship.

## References

- Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. 2011a. Analyzing temporal dynamics in Twitter profiles for personalized recommendations in the social web. In *Proceedings of the ACM Web Science Conference 2011*, pages 1–8, Koblenz, Germany.
- Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. 2011b. Analyzing user modeling on Twitter for personalized news recommendations. *User Modeling, Adaptation and Personalization*, pages 1–12.
- Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. 2011c. Semantic enrichment of twitter posts for user profile construction on the social web. *The Semantic Web: Research and Applications*, pages 375–389.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.
- Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual web search engine. *Proceedings of the 7th International Conference on World Wide Web*, 30(1-7):107–117.
- Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 335–336, Melbourne, Australia.
- Jilin Chen, Rowan Nairn, Les Nelson, Michael Bernstein, and Ed Chi. 2010. Short and tweet: experiments on recommending content from information streams. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems*, pages 1185–1194, Atlanta, Georgia.
- Junghoo Cho and Uri Schonfeld. 2007. Rankmass crawler: a crawler with high personalized pagerank coverage guarantee. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 375–386, Vienna, Austria.
- Anlei Dong, Ruiqiang Zhang, Pranam Kolari, Jing Bai, Fernando Diaz, Yi Chang, Zhaohui Zheng, and Hongyuan Zha. 2010. Time is of the essence: improving recency ranking using Twitter data. In *Proceedings of the 19th International Conference on World Wide Web*, pages 331–340, Raleigh, North Carolina.
- Yajuan Duan, Long Jiang, Tao Qin, Ming Zhou, and Heung-Yeung Shum. 2010. An empirical study on learning to rank of tweets. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 295–303, Beijing, China.
- John Hannon, Mike Bennett, and Barry Smyth. 2010. Recommending twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the 4th ACM Conference on Recommender Systems*, pages 199–206, Barcelona, Spain.
- Liangjie Hong, Ovidiu Dan, and Brian D. Davison. 2011. Predicting popular messages in Twitter. In *Proceedings of the 20th International Conference Companion on World Wide Web*, pages 57–58, Hyderabad, India.
- Minlie Huang, Yi Yang, and Xiaoyan Zhu. 2011. Quality-biased ranking of short texts in microblogging services. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 373–382, Chiang Mai, Thailand.
- Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20:422–446.
- Thorsten Joachims. 1999. Making large-scale svm learning practical. In *Advances in Kernel Methods: Support Vector Learning*, pages 169–184. MIT press.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*, volume 1. Cambridge University Press.
- Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1009–1018, Washington, DC.
- Emily Pitler, Annie Louis, and Ani Nenkova. 2010. Automatic evaluation of linguistic quality in multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 544–554, Uppsala, Sweden.
- Feng Qiu and Junghoo Cho. 2006. Automatic identification of user interest for personalized search. In *Proceedings of the 15th International Conference on World Wide Web*, pages 727–736, Edinburgh, Scotland.
- Sun Aaron R., Cheng Jiesi, Zeng, and Daniel Dajun. 2009. A novel recommendation framework for microblogging based on information diffusion. In *Proceedings of the 19th Annual Workshop on Information Technologies and Systems*, pages 199–216, Phoenix, Arizona.
- Daniel Ramage, Susan Dumais, and Dan Liebling. 2010. Characterizing microblogs with topic models. In *International AAAI Conference on Weblogs and Social Media*, pages 130–137. The AAAI Press.
- Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.
- Jaime Teevan, Daniel Ramage, and Meredith Ringel Morris. 2011. #Twittersearch: a comparison of microblog search and web search. In *Proceedings of the 4th ACM*



- International Conference on Web Search and Data Mining*, pages 35–44, Hong Kong, China.
- Ibrahim Uysal and W. Bruce Croft. 2011. User oriented tweet ranking: a filtering approach to microblogs. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 2261–2264, Glasgow, Scotland.
- Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Single document summarization with document expansion. In *Proceedings of the 22nd Conference on Artificial Intelligence*, pages 931–936, Vancouver, British Columbia.
- Xiaojun Wan, Huiying Li, and Jianguo Xiao. 2010. Cross-language document summarization based on machine translation quality prediction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 917–926, Uppsala, Sweden.
- Rui Yan, Jian-Yun Nie, and Xiaoming Li. 2011. Summarize what you are interested in: An optimization framework for interactive personalized summarization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1342–1351. Association for Computational Linguistics.
- Ding Zhou, Sergey A. Orshanskiy, Hongyuan Zha, and C. Lee Giles. 2007. Co-ranking authors and documents in a heterogeneous network. In *Proceedings of the 7th IEEE International Conference on Data Mining*, pages 739–744. IEEE.

# Joint Inference of Named Entity Recognition and Normalization for Tweets

Xiaohua Liu<sup>‡ †</sup>, Ming Zhou<sup>†</sup>, Furu Wei<sup>†</sup>, Zhongyang Fu<sup>§</sup>, Xiangyang Zhou<sup>#</sup>

<sup>‡</sup>School of Computer Science and Technology

Harbin Institute of Technology, Harbin, 150001, China

<sup>§</sup>Department of Computer Science and Engineering

Shanghai Jiao Tong University, Shanghai, 200240, China

<sup>#</sup>School of Computer Science and Technology

Shandong University, Jinan, 250100, China

<sup>†</sup>Microsoft Research Asia

Beijing, 100190, China

<sup>†</sup>{xiaoliu, fuwei, mingzhou}@microsoft.com

<sup>§</sup> zhongyang.fu@gmail.com <sup>#</sup> v-xzho@microsoft.com

## Abstract

Tweets represent a critical source of fresh information, in which named entities occur frequently with rich variations. We study the problem of named entity normalization (NEN) for tweets. Two main challenges are the errors propagated from named entity recognition (NER) and the dearth of information in a single tweet. We propose a novel graphical model to simultaneously conduct NER and NEN on multiple tweets to address these challenges. Particularly, our model introduces a binary random variable for each pair of words with the same lemma across similar tweets, whose value indicates whether the two related words are mentions of the same entity. We evaluate our method on a manually annotated data set, and show that our method outperforms the baseline that handles these two tasks separately, boosting the F1 from 80.2% to 83.6% for NER, and the Accuracy from 79.4% to 82.6% for NEN, respectively.

## 1 Introduction

Tweets, short messages of less than 140 characters shared through the Twitter service<sup>1</sup>, have become an important source of fresh information. As a result, the task of named entity recognition (NER) for tweets, which aims to identify mentions of rigid designators from tweets belonging to named-entity types such as persons, organizations and locations (2007), has attracted increasing research interest. For example, Ritter et al. (2011) develop a system that exploits a CRF model to segment named

entities and then uses a distantly supervised approach based on LabeledLDA to classify named entities. Liu et al. (2011) combine a classifier based on the k-nearest neighbors algorithm with a CRF-based model to leverage cross tweets information, and adopt the semi-supervised learning to leverage unlabeled tweets.

However, named entity normalization (NEN) for tweets, which transforms named entities mentioned in tweets to their unambiguous canonical forms, has not been well studied. Owing to the informal nature of tweets, there are rich variations of named entities in them. According to our investigation on the data set provided by Liu et al. (2011), every named entity in tweets has an average of 3.3 variations<sup>2</sup>. As an illustrative example, we show “Anneke Gronloh”, which may occur as “Mw.,Gronloh”, “Anneke Kronloh” or “Mevrouw G”. We thus propose NEN for tweets, which plays an important role in entity retrieval, trend detection, and event and entity tracking. For example, Khalid et al. (2008) show that even a simple normalization method leads to improvements of early precision, for both document and passage retrieval, and better normalization results in better retrieval performance.

Traditionally, NEN is regarded as a septated task, which takes the output of NER as its input (Li et al., 2002; Cohen, 2005; Jijkoun et al., 2008; Dai et al., 2011). One limitation of this cascaded approach is that errors propagate from NER to NEN and there is no feedback from NEN to NER. As demonstrated by Khalid et al. (2008), most NEN errors are caused

<sup>2</sup>This data set consists of 12,245 randomly sampled tweets within five days.

<sup>1</sup><http://www.twitter.com>

by recognition errors. Another challenge of NEN is the dearth of information in a single tweet, due to the short and noise-prone nature of tweets. Reportedly, the accuracy of a baseline NEN system based on Wikipedia drops considerably from 94% on edited news to 77% on news comments, a kind of user generated content (UGC) with similar style to tweets (Jijkoun et al., 2008).

We propose jointly conducting NER and NEN on multiple tweets using a graphical model, to address these challenges. Intuitively, improving the performance of NER boosts the performance of NEN. For example, consider the following two tweets: “... Alex’s jokes. Justin’s smartness. Max’s randomness...” and “... Alex Russo was like the best character on Disney Channel...”. Identifying “Alex” and “Alex Russo” as PERSON will encourage NEN systems to normalize “Alex” into “Alex Russo”. On the other hand, NEN can guide NER. For instance, consider the following two tweets: “... she knew Burger King when he was a Prince!...” and “... I’m craving all sorts of food: mcdonalds, burger king, pizza, chinese...”. Suppose the NEN system believes that “burger king” cannot be mapped to “Burger King” since these two tweets are not similar in content. This will help NER to assign them different types of labels. Our method optimizes these two tasks simultaneously by enabling them to interact with each other. This largely differentiates our method from existing work.

Furthermore, considering multiple tweets simultaneously allows us to exploit the redundancy in tweets, as suggested by Liu et al. (2011). For example, consider the following two tweets: “... Bobby Shaw you don’t invite the wind...” and “... I own yah ! Lool bobby shaw...”. Recognizing “Bobby Shaw” in the first tweet as a PERSON is easy owing to its capitalization and the following word “you”, which in turn helps to identify “bobby shaw” in the second tweet as a PERSON.

We adopt a factor graph as our graphical model, which is constructed in the following manner. We first introduce a random variable for each word in every tweet, which represents the BILOU (Beginning, the Inside and the Last tokens of multi-token entities as well as Unit-length entities) label of the corresponding word. Then we add a factor to connect two neighboring variables, forming a conven-

tional linear chain CRFs. Hereafter, we use  $t_m$  to denote the  $m^{th}$  tweet,  $t_m^i$  and  $y_m^i$  to denote the  $i^{th}$  word of  $t_m$  and its BILOU label, respectively, and  $f_m^i$  to denote the factor related to  $y_m^{i-1}$  and  $y_m^i$ . Next, for each word pair with the same lemma, denoted by  $t_m^i$  and  $t_n^j$ , we introduce a binary random variable, denoted by  $z_{mn}^{ij}$ , whose value indicates whether  $t_m^i$  and  $t_n^j$  belong to two mentions of the same entity. Finally, for any  $z_{mn}^{ij}$  we add a factor, denoted by  $f_{mn}^{ij}$ , to connect  $y_m^i$ ,  $y_n^j$  and  $z_{mn}^{ij}$ . Factors in the same group ( $\{f_{mn}^{ij}\}$  or  $\{f_m^i\}$ ) share the same set of feature templates. Figure 1 illustrates an example of our factor graph for two tweets.

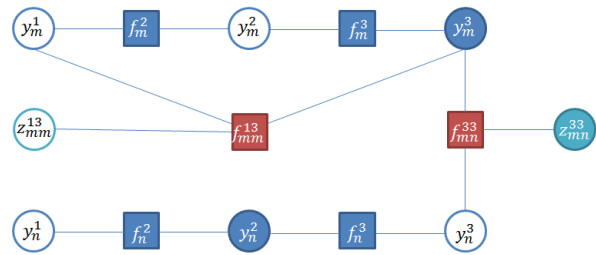


Figure 1: A factor graph that jointly conducts NER and NEN on multiple tweets. Blue and green circles represent NE type ( $y$ -serials) and normalization variables ( $z$ -serials), respectively; filled circles indicate observed random variables; blue rectangles represent the factors connecting neighboring  $y$ -serial variables while red rectangles stand for the factors connecting distant  $y$ -serial and  $z$ -serial variables.

It is worth noting that our factor graph is different from the skip-chain CRFs (Galley, 2006) in the sense that any skip-chain factor of our model consists not only of two NE type variables ( $y_m^i$  and  $y_n^j$ ), which is the case for skip-chain CRFs, but also a normalization variable ( $z_{mn}^{ij}$ ). It is these normalization variables that enable us to conduct NER and NEN jointly.

We manually add normalization information to the data set shared by Liu et al. (2011), to evaluate our method. Experimental results show that our method achieves 83.6% F1 for NER and 82.6% Accuracy for NEN, outperforming the baseline with 80.2%F1 for NER and 79.4% Accuracy for NEN.

We summarize our contributions as follows.

1. We introduce the task of NEN for tweets, and propose jointly conducting NER and NEN for

multiple tweets using a factor graph, which leverages redundancy in tweets to make up for the dearth of information in a single tweet and allows these two tasks to inform each other.

2. We evaluate our method on a human annotated data set, and show that our method compares favorably with the baseline, achieving better performance in both tasks.

Our paper is organized as follows. In the next section, we introduce related work. In Section 3 and 4, we formally define the task and present our method. In Section 5, we evaluate our method. And finally we conclude our work in Section 6.

## 2 Related Work

Related work can be divided into two categories: NER and NEN.

### 2.1 NER

NER has been well studied and its solutions can be divided into three categories: 1) Rule-based (Krupka and Hausman, 1998); 2) machine learning based (Finkel and Manning, 2009; Singh et al., 2010); and 3) hybrid methods (Jansche and Abney, 2002). Owing to the availability of annotated corpora, such as ACE05, Enron (Minkov et al., 2005) and CoNLL03 (Tjong Kim Sang and De Meulder, 2003), data driven methods are now dominant.

Current studies of NER mainly focus on formal text such as news articles (Mccallum and Li, 2003; Etzioni et al., 2005). A representative work is that of Ratinov and Roth (2009), in which they systematically study the challenges of NER, compare several solutions, and show some interesting findings. For example, they show that the BILOU encoding scheme significantly outperforms the BIO schema (Beginning, the Inside and Outside of a chunk).

A handful of work on other genres of texts exists. For example, Yoshida and Tsujii build a biomedical NER system (2007) using lexical features, orthographic features, semantic features and syntactic features, such as part-of-speech (POS) and shallow parsing; Downey et al. (2007) employ capitalization cues and n-gram statistics to locate names of a variety of classes in web text; Wang (2009) introduces NER to clinical notes. A linear CRF model

is trained on a manually annotated data set, which achieves an F1 of 81.48% on the test data set; Chiticariu et al. (2010) design and implement a high-level language NERL which simplifies the process of building, understanding, and customizing complex rule-based named-entity annotators for different domains.

Recently, NER for Tweets attracts growing interest. Finin et al. (2010) use Amazons Mechanical Turk service<sup>3</sup> and CrowdFlower<sup>4</sup> to annotate named entities in tweets and train a CRF model to evaluate the effectiveness of human labeling. Ritter et al. (2011) re-build the NLP pipeline for tweets beginning with POS tagging, through chunking, to NER, which first exploits a CRF model to segment named entities and then uses a distantly supervised approach based on LabeledLDA to classify named entities. Unlike this work, our work detects the boundary and type of a named entity simultaneously using sequential labeling techniques. Liu et al. (2011) combine a classifier based on the k-nearest neighbors algorithm with a CRF-based model to leverage cross tweets information, and adopt the semi-supervised learning to leverage unlabeled tweets. Our method leverages redundancy in similar tweets, using a factor graph rather than a two-stage labeling strategy. One advantage of our method is that local and global information can interact with each other.

### 2.2 NEN

There is a large body of studies into normalizing various types of entities for formally written texts. For instance, Cohen (2005) normalizes gene/protein names using dictionaries automatically extracted from gene databases; Magdy et al. (2007) address cross-document Arabic name normalization using a machine learning approach, a dictionary of person names and frequency information for names in a collection; Cucerzan (2007) demonstrates a large-scale system for the recognition and semantic disambiguation of named entities based on information extracted from a large encyclopedic collection and Web search results; Dai et al. (2011) employ a Markov logic network to model interweaved con-

<sup>3</sup><https://www.mturk.com/mturk/>

<sup>4</sup><http://crowdfunder.com/>

straints in a setting of gene mention normalization.

Jijkoun et al. (2008) study NEN for UGC. They report that the accuracy of a baseline NEN system based on Wikipedia drops considerably from 94% on edited news to 77% on UGC. They identify three main error sources, i.e., entity recognition errors, multiple ways of referring to the same entity and ambiguous references, and exploit hand-crafted rules to improve the baseline NEN system.

We introduce the task of NEN for tweets, a new genre of texts with rich entity variations. In contrast to existing NEN systems, which take the output of NER systems as their input, our method conducts NER and NEN at the same time, allowing them to reinforce each other, as demonstrated by the experimental results.

### 3 Task Definition

A tweet is a short text message with no more than 140 characters. Here is an example of a tweet: “mycraftingworld: #Win Microsoft Office 2010 Home and Student #Contest from @office http://bit.ly/...”, where “mycraftingworld” is the name of the user who published this tweet. Words beginning with “#” like “#Win” are hash tags; words starting with “@” like “@office” represent user names; and “http://bit.ly/” is a shortened link.

Given a set of tweets, e.g., tweets within some period or related to some query, our task is: 1) To recognize each mention of entities of predefined types for each tweet; and 2) to restore each entity mention into its unambiguous canonical form. Following Liu et al. (2011), we focus on four types of entities, i.e., PERSON, ORGANIZATION, PRODUCT, and LOCATION, and constrain our scope to English tweets. Note that the NEN sub-task can be transformed as follows. Given each pair of entity mentions, decide whether they denote the same entity. Once this is achieved, we can link all the mentions of the same entity, and choose a representative mention, e.g., the longest mention, as their canonical form.

As an illustrative example, consider the following three tweets: “...Gaga’s Christmas dinner with her family. Awwwwwn...”, “...Lady Gaaaaga with her family on Christmas...” and “...Buying a magazine just because Lady Gaga’s on the cover...”. It is expected that “Gaga”, “Lady Gaaaaga” and “Lady

Gaga” are all labeled as PERSON, and can be restored as “Lady Gaga”.

## 4 Our Method

In contrast to existing work, our method jointly conducts NER and NEN for multiple tweets. We first give an overview of our method, then detail its model and features.

### 4.1 Overview

Given a set of tweets as input, our method recognizes predefined types of named entities and for each entity outputs its unambiguous canonical form.

To resolve NER, we assign a label to each word in a tweet, indicating both the boundary and entity type. Following Ratnov and Roth (2009), we use the BILOU schema. For example, consider the tweet “...without you is like an iphone without apps; Lady gaga without her telephone...”, the labeled sequence using the BILOU schema is: “...without<sub>O</sub> you<sub>O</sub> is<sub>O</sub> like<sub>O</sub> an<sub>O</sub> iphone<sub>U-PRODUCT</sub> without<sub>O</sub> apps<sub>O</sub>; Lady<sub>B-PERSON</sub> gaga<sub>L-PERSON</sub> without<sub>O</sub> her<sub>O</sub> telephone<sub>O</sub>...”, where “iphone<sub>U-PRODUCT</sub>” indicates that “iphone” is a product name of unit length; “Lady<sub>B-PERSON</sub>” means “Lady” is the beginning of a person name while “gaga<sub>L-PERSON</sub>” suggests that “gaga” is the last token of a person name.

To resolve NEN, we assign a binary value label  $z_{mn}^{ij}$  to each pair of words  $t_m^i$  and  $t_n^j$  which share the same lemma.  $z_{mn}^{ij} = 1$  or  $-1$ , indicating whether  $t_m^i$  and  $t_n^j$  belong to two mentions of the same entity<sup>5</sup>. For example, consider the three tweets presented in Section 3. “Gaga<sub>1</sub>”<sup>6</sup> and “Gaga<sub>3</sub>” will be assigned a “1” label, since they are part of two mentions of the same entity “Lady Gaga”; similarly, “Lady<sub>2</sub>” and “Lady<sub>3</sub>” are connected with a “1” label. Note that there are no NEN labels for pairs like “her<sub>1</sub>” and “her<sub>2</sub>” or “with<sub>1</sub>” and “with<sub>2</sub>”, since words like “her” and “with” are stop words.

With NE type and normalization labels obtained, we judge two mentions, denoted by  $t_m^{i_1 \dots i_k}$  and

<sup>5</sup>Stop words have no normalization labels. The stop words are mainly from <http://www.textfixer.com/resources/common-english-words.txt>.

<sup>6</sup>We use  $w_m^i$  to denote word  $w$ ’s  $i^{th}$  appearance in the  $m^{th}$  tweet. For example, “Gaga<sub>1</sub>” denotes the first occurrence of “Gaga” in the first tweet.

$t_n^{j_1 \dots j_l}$ , respectively, refer to the same entity if and only if: 1) The two mentions share the same entity type; 2)  $t_m^{i_1 \dots i_k}$  is a sub-string of  $t_n^{j_1 \dots j_l}$  or vice versa; and 3)  $z_{mn}^{ij} = 1$ ,  $i = i_1, \dots, i_k$  and  $j = j_1, \dots, j_l$ , if  $z_{mn}^{ij}$  exists. Still take the three tweets presented in Section 3 for example. Suppose ‘‘Gaga<sub>1</sub><sup>1</sup>’’ and ‘‘Lady Gaga<sub>3</sub><sup>1</sup>’’ are labeled as PERSON, and there is only one related NE normalization label, which is associated with ‘‘Gaga<sub>1</sub><sup>1</sup>’’ and ‘‘Gaga<sub>3</sub><sup>1</sup>’’ and has 1 as its value. We then consider that these two mentions can be normalized into the same entity; in a similar way, we can align ‘‘Lady<sub>2</sub><sup>1</sup> Gaaaaga’’ with ‘‘Lady<sub>3</sub><sup>1</sup> Gaga’’. Combining these pieces information together, we can infer that ‘‘Gaga<sub>1</sub><sup>1</sup>’’, ‘‘Lady<sub>2</sub><sup>1</sup> Gaaaaga’’ and ‘‘Lady<sub>3</sub><sup>1</sup> Gaga’’ are three mentions of the same entity. Finally, we can select ‘‘Lady<sub>3</sub><sup>1</sup> Gaga’’ as the representative, and output ‘‘Lady Gaga’’ as their canonical form. We choose the mention with the maximum number of words as the representative. In case of a tie, we prefer the mention with an Wikipedia entry <sup>7</sup>.

The central problem with our method is inferring all the NE type (*y*-serial) and normalization (*z*-serial) variables. To achieve this, we construct a factor graph according to the input tweets, which can evaluate the probability of every possible assignment of *y*-serials and *z*-serials, by checking the characteristics of the assignment. Each characteristic is called a feature. In this way, we can select the assignment with the highest probability. Next we will introduce our model in detail, including its training and inference procedure and features.

## 4.2 Model

We adopt a factor graph as our model. One advantage of our model is that it allows *y*-serials and *z*-serials variables to interact with each other to jointly optimize NER and NEN.

Given a set of tweets  $T = \{t_m\}_{m=1}^N$ , we can build a factor graph  $\mathcal{G} = (Y, Z, F, E)$ , where:  $Y$  and  $Z$  denote *y*-serials and *z*-serials variables, respectively;  $F$  represents factor vertices, consisting of  $\{f_m^i\}$  and  $\{f_{mn}^{ij}\}$ ,  $f_m^i = f_m^i(y_m^{i-1}, y_m^i)$  and  $f_{mn}^{ij} = f_{mn}^{ij}(y_m^i, y_n^j, z_{mn}^{ij})$ ;  $E$  stands for edges, which depends on  $F$ , and consists of edges between  $y_m^{i-1}$  and  $y_m^i$ , and those between  $y_m^i, y_n^j$  and  $f_{mn}^{ij}$ .

<sup>7</sup>If it still ends up as a draw, we will randomly choose one from the best.

$\mathcal{G} = (Y, Z, F, E)$  defines a probability distribution according to Formula 1.

$$\ln P(Y, Z | \mathcal{G}, T) \propto \sum_{m,i} \ln f_m^i(y_m^{i-1}, y_m^i) + \sum_{m,n,i,j} \delta_{mn}^{ij} \cdot \ln f_{mn}^{ij}(y_m^i, y_n^j, z_{mn}^{ij}) \quad (1)$$

where  $\delta_{mn}^{ij} = 1$  if and only if  $t_m^i$  and  $t_n^j$  have the same lemma and are not stop words, otherwise zero. A factor factorizes according to a set of features, so that:

$$\begin{aligned} \ln f_m^i(y_m^{i-1}, y_m^i) &= \sum_k \lambda_k^{(1)} \phi_k^{(1)}(y_m^{i-1}, y_m^i) \\ \ln f_{mn}^{ij}(y_m^i, y_n^j, z_{mn}^{ij}) &= \sum_k \lambda_k^{(2)} \phi_k^{(2)}(y_m^i, y_n^j, z_{mn}^{ij}) \end{aligned} \quad (2)$$

$\{\phi_k^{(1)}\}_{k=1}^{K_1}$  and  $\{\phi_k^{(2)}\}_{k=1}^{K_2}$  are two feature sets.  $\Theta = \{\lambda_k^{(1)}\}_{k=1}^{K_1} \cup \{\lambda_k^{(2)}\}_{k=1}^{K_2}$  is called the feature weight set or parameter set of  $\mathcal{G}$ . Each feature has a real value as its weight.

**Training**  $\Theta$  is learnt from annotated tweets  $T$ , by maximizing the data likelihood, i.e.,

$$\Theta^* = \arg \max_{\Theta} \ln P(Y, Z | \Theta, T) \quad (3)$$

To solve this optimization problem, we first calculate its gradient:

$$\begin{aligned} \frac{\partial \ln P(Y, Z | T; \Theta)}{\partial \lambda_k^{(1)}} &= \sum_{m,i} \phi_k^{(1)}(y_m^{i-1}, y_m^i) \\ &- \sum_{m,i} \sum_{y_m^{i-1}, y_m^i} p(y_m^{i-1}, y_m^i | T; \Theta) \phi_k^{(1)}(y_m^{i-1}, y_m^i) \end{aligned} \quad (4)$$

$$\begin{aligned} \frac{\partial \ln P(Y, Z | T; \Theta)}{\partial \lambda_k^{(2)}} &= \sum_{m,n,i,j} \delta_{mn}^{ij} \cdot \phi_k^{(2)}(y_m^i, y_n^j, z_{mn}^{ij}) \\ &- \sum_{m,n,i,j} \delta_{mn}^{ij} \sum_{y_m^i, y_n^j, z_{mn}^{ij}} p(y_m^i, y_n^j, z_{mn}^{ij} | T; \Theta) \\ &\quad \cdot \phi_k^{(2)}(y_m^i, y_n^j, z_{mn}^{ij}) \end{aligned} \quad (5)$$

Here, the two marginal probabilities  $p(y_m^{i-1}, y_m^i | T; \Theta)$  and  $p(y_m^i, y_n^j, z_{mn}^{ij} | T; \Theta)$  are computed using loopy belief propagation (Murphy et al., 1999). Once we have computed the gradient,  $\Theta^*$  can be worked out by standard techniques such as steepest descent, conjugate gradient and the

limited-memory BFGS algorithm (L-BFGS). We choose L-BFGS because it is particularly well suited for optimization problems with a large number of variables.

**Inference** Supposing the parameters  $\Theta$  have been set to  $\Theta^*$ , the inference problem is: Given a set of testing tweets  $T$ , output the most probable assignment of  $Y$  and  $Z$ , i.e.,

$$(Y, Z)^* = \arg \max_{(Y, Z)} \ln P(Y, Z | \Theta^*, T) \quad (6)$$

We adopt the max-product algorithm to solve this inference problem. The max-product algorithm is nearly identical to the loopy belief propagation algorithm, with the sums replaced by maxima in the definitions. Note that in both the training and testing stage, the factor graph is constructed in the same way as described in Section 1.

**Efficiency** We take several actions to improve our model’s efficiency. Firstly, we manually compile a comprehensive named entity dictionary from various sources including Wikipedia, Freebase<sup>8</sup>, news articles and the gazetteers shared by Ratnov and Roth (2009). In total this dictionary contains 350 million entries<sup>9</sup>. By looking up this dictionary<sup>10</sup>, we generate the possible BILOU labels, denoted by  $Y_m^i$  hereafter, for each word  $t_m^i$ . For instance, consider “... Good Morning new<sub>1</sub><sup>1</sup> york<sub>1</sub><sup>1</sup>...”. Suppose “New York City” and “New York Times” are in our dictionary, then “new<sub>1</sub><sup>1</sup> york<sub>1</sub><sup>1</sup>” is the matched string with two corresponding entities. As a result, “B-LOCATION” and “B-ORGANIZATION” will be added to  $Y_{new_1^1}$ , and “I-LOCATION” and “I-ORGANIZATION” will be added to  $Y_{york_1^1}$ . If  $Y_m^i \neq \emptyset$ , we enforce the constraint for training and testing that  $y_m^i \in Y_m^i$ , to reduce the search space.

Secondly, in the testing phase, we introduce three rules related to  $z_{mn}^{ij}$ : 1)  $z_{mm}^{ij} = 1$ , which says two words sharing the same lemma in the same tweet denote the same entity; 2) set  $z_{mn}^{ij}$  to 1, if the similarity between  $t_m$  and  $t_n$  is above a threshold (0.8 in our work), or  $t_m$  and  $t_n$  share one hash tag; and 3)  $z_{mn}^{ij} = -1$ , if the similarity between  $t_m$  and  $t_n$  is below a threshold (0.3 in work). To compute

<sup>8</sup><http://freebase.com/view/military>

<sup>9</sup>One phrase referring to  $L$  entities has  $L$  entries.

<sup>10</sup>We use case-insensitive leftmost longest match.

the similarity, each tweet is represented as a bag-of-words vector with the stop words removed, and the cosine similarity is adopted, as defined in Formula 7. These rules pre-label a significant part of  $z$ -serial variables (accounting for 22.5%), with an accuracy of 93.5%.

$$sim(t_m, t_n) = \frac{\vec{t}_m \cdot \vec{t}_n}{|\vec{t}_m| |\vec{t}_n|} \quad (7)$$

Note that in our experiments, these measures reduce the training and testing time by 36.2% and 62.8%, respectively, while no obvious performance drop is observed.

### 4.3 Features

A feature in  $\{\phi_k^{(1)}\}_{k=1}^{K_1}$  involves a pair of neighboring NE-type labels, i.e.,  $y_m^{i-1}$  and  $y_m^i$ , while a feature in  $\{\phi_k^{(2)}\}_{k=1}^{K_2}$  concerns a pair of distant NE-type labels and its associated normalization label, i.e.,  $y_m^i, y_n^j$  and  $z_{mn}^{ij}$ . Details are given below.

#### 4.3.1 Feature Set One: $\{\phi_k^{(1)}\}_{k=1}^{K_1}$

We adopts features similar to Wang (2009), and Ratnov and Roth (2009), i.e., orthographic features, lexical features and gazetteer-related features. These features are defined on the observation. Combining them with  $y_m^{i-1}$  and  $y_m^i$  constitutes  $\{\phi_k^{(1)}\}_{k=1}^{K_1}$ .

**Orthographic features:** Whether  $t_m^i$  is capitalized or upper case; whether it is alphanumeric or contains any slashes; whether it is a stop word; word prefixes and suffixes.

**Lexical features:** Lemma of  $t_m^i$ ,  $t_m^{i-1}$  and  $t_m^{i+1}$ , respectively; whether  $t_m^i$  is an out-of-vocabulary (OOV) word<sup>11</sup>; POS of  $t_m^i$ ,  $t_m^{i-1}$  and  $t_m^{i+1}$ , respectively; whether  $t_m^i$  is a hash tag, a link, or a user account.

**Gazetteer-related features:** Whether  $Y_m^i$  is empty; the dominating label/entity type in  $Y_m^i$ . Which one is dominant is decided by majority voting of the entities in our dictionary. In case of a tie, we randomly choose one from the best.

#### 4.3.2 Feature Set Two: $\{\phi_k^{(2)}\}_{k=1}^{K_2}$

Similarly, we define orthographic, lexical features and gazetteer-related features on the observation,  $y_m^i$

<sup>11</sup>We first conduct a simple dictionary-lookup based normalization with the incorrect/correct word pair list provided by Han et al. (2011) to correct common ill-formed words. Then we call an online dictionary service to judge whether a word is OOV.

and  $y_n^j$ ; and then we combine these features with  $z_{mn}^{ij}$ , forming  $\{\phi_k^{(2)}\}_{k=1}^{K_2}$ .

**Orthographic features:** Whether  $t_m^i / t_n^j$  is capitalized or upper case; whether  $t_m^i / t_n^j$  is alphanumeric or contains any slashes; prefixes and suffixes of  $t_m^i$ .

**Lexical features:** Lemma of  $t_m^i$ ; whether  $t_m^i$  is OOV; whether  $t_m^i / t_m^{i+1} / t_m^{i-1}$  and  $t_n^j / t_n^{j+1} / t_n^{j-1}$  have the same POS; whether  $y_m^i$  and  $y_n^j$  have the same label/entity type.

**Gazetteer-related features:** Whether  $Y_m^i \cap Y_n^j / Y_m^{i+1} \cap Y_n^{j+1} / Y_m^{i-1} \cap Y_n^{j-1}$  is empty; whether the dominating label/entity type in  $Y_m^i$  is the same as that in  $Y_n^j$ .

## 5 Experiments

We manually annotate a data set to evaluate our method. We show that our method outperforms the baseline, a cascaded system that conducts NER and NEN individually.

### 5.1 Data Preparation

We use the data set provided by Liu et al. (2011), which consists of 12,245 tweets with four types of entities annotated: PERSON, LOCATION, ORGANIZATION and PRODUCT. We enrich this data set by adding entity normalization information. Two annotators<sup>12</sup> are involved. For any entity mention, two annotators independently annotate its canonical form. The inter-rater agreement measured by kappa is 0.72. Any inconsistent case is discussed by the two annotators till a consensus is reached. 2,245 tweets are used for development, and the remainder are used for 5-fold cross validation.

### 5.2 Evaluation Metrics

We adopt the widely-used Precision, Recall and F1 to measure the performance of NER for a particular type of entity, and the average Precision, Recall and F1 to measure the overall performance of NER (Liu et al., 2011; Ritter et al., 2011). As for NEN, we adopt the widely-used Accuracy, i.e., to what percentage the outputted canonical forms are correct (Jijkoun et al., 2008; Cucerzan, 2007; Li et al., 2002).

<sup>12</sup>Two native English speakers.

### 5.3 Baseline

We develop a cascaded system as the baseline, which conducts NER and NEN sequentially. Its NER module, denoted by  $S_{BR}$ , is based on the state-of-the-art method introduced by Liu et al. (2011); and its NEN model, denoted by  $S_{BN}$ , follows the NEN system for user-generated news comments proposed by Jijkoun et al. (2008), which uses handcrafted rules to improve a typical NEN system that normalizes surface forms to Wikipedia page titles. We use the POS tagger developed by Ritter et al. (2011) to extract POS related features, and the OpenNLP toolkit to get lemma related features.

### 5.4 Results

Tables 1- 2 show the overall performance of the baseline and ours (denoted by  $S_{RN}$ ). It can be seen that, our method yields a significantly higher F1 (with  $p < 0.01$ ) than  $S_{BR}$ , and a moderate improvement of accuracy as compared with  $S_{BN}$  (with  $p < 0.05$ ). As a case study, we show that our system successfully identified “jaxon<sub>1</sub>” as a PERSON in the tweet “... come to see jaxon<sub>1</sub> someday...”, which is mistakenly labeled as a LOCATION by  $S_{BR}$ . This is largely owing to the fact that our system aligns “jaxon<sub>1</sub>” with “Jaxson<sub>2</sub>” in the tweet “... I love Jaxson<sub>2</sub>, Hes like my little brother...”, in which “Jaxson<sub>2</sub>” is identified as a PERSON. As a result, this encourages our system to consider “jaxon<sub>1</sub>” as a PERSON. We also find cases where our system works but  $S_{BN}$  fails. For example, “Goldman<sub>1</sub>” in the tweet “... Goldman sees massive upside risk in oil prices...” is normalized into “Albert Goldman” by  $S_{BR}$ , because it is mistakenly identified as a PERSON by  $S_{BR}$ ; in contrast, our system recognizes “Goldman<sub>2</sub> Sachs” as an ORGANIZATION, and successfully links “Goldman<sub>2</sub>” to “Goldman<sub>1</sub>”, resulting that “Goldman<sub>1</sub>” is identified as an ORGANIZATION and normalized into “Goldman Sachs”.

Table 3 reports the NER performance of our method for each entity type, from which we see that our system consistently yields better F1 on all entity types than  $S_{BR}$ . We also see that our system boosts the F1 for ORGANIZATION most significantly, reflecting the fact that a large number of organizations that are incorrectly labeled as PERSON by  $S_{BR}$ , are now correctly recognized by our method.



System	Pre	Rec	F1
$S_{RN}$	84.7	82.5	83.6
$S_{BR}$	81.6	78.8	80.2

Table 1: Overall performance (%) of NER.

System	Accuracy
$S_{RN}$	82.6
$S_{BN}$	79.4

Table 2: Overall Accuracy (%) of NEN .

System	PER	PRO	LOC	ORG
$S_{RN}$	84.2	80.5	82.1	85.2
$S_{BR}$	83.9	78.7	81.3	79.8

Table 3: F1 (%) of NER on different entity types.

Features	NER (F1)	NEN (Accuracy)
$F_o$	59.2	61.3
$F_o + F_l$	65.8	68.7
$F_o + F_g$	80.1	77.2
$F_o + F_l + F_g$	83.6	82.6

Table 4: Overall F1 (%) of NER and Accuracy (%) of NEN with different feature sets.

Table 4 shows the overall performance of our method with various feature set combinations, where  $F_o$ ,  $F_l$  and  $F_g$  denote the orthographic features, the lexical features, and the gazetteer-related features, respectively. From Table 4 we see that gazetteer-related features significantly boost the F1 for NER and Accuracy for NEN, suggesting the importance of external knowledge for this task.

## 5.5 Discussion

One main error source for NER and NEN, which accounts for more than half of all the errors, is slang expressions and informal abbreviations. For instance, our method recognizes “California<sub>1</sub>” in the tweet “... And Now, He Lives All The Way In California<sub>1</sub>...” as a LOCATION, however, it mistakenly identifies “Cali<sub>2</sub>” in the tweet “... i love Cali so much...” as a PERSON. One reason is our system does not generate any *z*-serial variable for “California<sub>1</sub>” and “Cali<sub>2</sub>” since they have different lemmas. A more complicated case is “BS<sub>1</sub>” in the tweet “... I, bobby shaw, am gonna put BS<sub>1</sub> on

everything...”, in which “BS<sub>1</sub>” is the abbreviation of “bobby shaw”. Our method fails to recognize “BS<sub>1</sub>” as an entity. There are two possible ways to fix these errors: 1) Extending the scope of *z*-serial variables to each word pairs with a common prefix; and 2) developing advanced normalization components to restore such slang expressions and informal abbreviations into their canonical forms.

Our method does not directly exploit Wikipedia for NEN. This explains the cases where our system correctly links multiple entity mentions but fails to generate canonical forms. Take the following two tweets for example: “... nitip link win7<sub>1</sub> sp1...” and “... Hit the 3TB wall on SRT installing fresh Win7<sub>2</sub>...”. Our system recognizes “win7<sub>1</sub>” and “Win7<sub>2</sub>” as two mentions of the same product, but cannot output their canonical forms “Windows 7”. One possible solution is to exploit Wikipedia to compile a dictionary consisting of entities and their variations.

## 6 Conclusions and Future work

We study the task of NEN for tweets, a new genre of texts that are short and prone to noise. Two challenges of this task are the dearth of information in a single tweet and errors propagated from the NER component. We propose jointly conducting NER and NEN for multiple tweets using a factor graph, to address these challenges. One unique characteristic of our model is that a NE normalization variable is introduced to indicate whether a word pair belongs to the mentions of the same entity. We evaluate our method on a manually annotated data set. Experimental results show our method yields better F1 for NER and Accuracy for NEN than the state-of-the-art baseline that conducts two tasks sequentially.

In the future, we plan to explore two directions to improve our method. First, we are going to develop advanced tweet normalization technologies to resolve slang expressions and informal abbreviations. Second, we are interested in incorporating knowledge mined from Wikipedia into our factor graph.

## Acknowledgments

We thank Yunbo Cao, Dongdong Zhang, and Mu Li for helpful discussions, and the anonymous reviewers for their valuable comments.

## References

- Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. Domain adaptation of rule-based annotators for named-entity recognition tasks. In *EMNLP*, pages 1002–1012.
- Aaron Cohen. 2005. Unsupervised gene/protein named entity normalization using automatically extracted dictionaries. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 17–24, Detroit, June. Association for Computational Linguistics.
- Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *In Proc. 2007 Joint Conference on EMNLP and CNLL*, pages 708–716.
- Hong-Jie Dai, Richard Tzong-Han Tsai, and Wen-Lian Hsu. 2011. Entity disambiguation using a markov-logic network. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 846–855, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Doug Downey, Matthew Broadhead, and Oren Etzioni. 2007. Locating Complex Named Entities in Web Text. In *IJCAI*.
- Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165(1):91–134.
- Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowd-sourcing. In *CSLDAMT*, pages 80–88.
- Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *EMNLP*, pages 141–150.
- Michel Galley. 2006. A skip-chain conditional random field for ranking meeting utterances by importance. In *Association for Computational Linguistics*, pages 364–372.
- Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *ACL HLT*.
- Martin Jansche and Steven P. Abney. 2002. Information extraction from voicemail transcripts. In *EMNLP*, pages 320–327.
- Valentin Jijkoun, Mahboob Alam Khalid, Maarten Marx, and Maarten de Rijke. 2008. Named entity normalization in user generated content. In *Proceedings of the second workshop on Analytics for noisy unstructured text data, AND '08*, pages 23–30, New York, NY, USA. ACM.
- Mahboob Khalid, Valentin Jijkoun, and Maarten de Rijke. 2008. The impact of named entity normalization on information retrieval for question answering. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen White, editors, *Advances in Information Retrieval*, volume 4956 of *Lecture Notes in Computer Science*, pages 705–710. Springer Berlin / Heidelberg.
- George R. Krupka and Kevin Hausman. 1998. Isoquest: Description of the netowl<sup>TM</sup> extractor system as used in muc-7. In *MUC-7*.
- Huifeng Li, Rohini K. Srihari, Cheng Niu, and Wei Li. 2002. Location normalization for information extraction. In *COLING*.
- Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *ACL*.
- Walid Magdy, Kareem Darwish, Ossama Emam, and Hany Hassan. 2007. Arabic cross-document person name normalization. In *In CASL Workshop 07*, pages 25–32.
- Andrew McCallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *HLT-NAACL*, pages 188–191.
- Einat Minkov, Richard C. Wang, and William W. Cohen. 2005. Extracting personal names from email: applying named entity recognition to informal text. In *HLT*, pages 443–450.
- Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *In Proceedings of Uncertainty in AI*, pages 467–475.
- David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30:3–26.
- Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155.
- Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Sameer Singh, Dustin Hillard, and Chris Leggetter. 2010. Minimally-supervised extraction of entities from text advertisements. In *HLT-NAACL*, pages 73–81.
- Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *HLT-NAACL*, pages 142–147.

- Yefeng Wang. 2009. Annotating and recognising named entities in clinical notes. In *ACL-IJCNLP*, pages 18–26.
- Kazuhiro Yoshida and Jun'ichi Tsujii. 2007. Reranking for biomedical named-entity recognition. In *BioNLP*, pages 209–216.

# Finding Bursty Topics from Microblogs

Qiming Diao, Jing Jiang, Feida Zhu, Ee-Peng Lim

Living Analytics Research Centre

School of Information Systems

Singapore Management University

{qiming.diao.2010, jingjiang, fdzhu, eplim}@smu.edu.sg

## Abstract

Microblogs such as Twitter reflect the general public's reactions to major events. Bursty topics from microblogs reveal what events have attracted the most online attention. Although bursty event detection from text streams has been studied before, previous work may not be suitable for microblogs because compared with other text streams such as news articles and scientific publications, microblog posts are particularly diverse and noisy. To find topics that have bursty patterns on microblogs, we propose a topic model that simultaneously captures two observations: (1) posts published around the same time are more likely to have the same topic, and (2) posts published by the same user are more likely to have the same topic. The former helps find event-driven posts while the latter helps identify and filter out "personal" posts. Our experiments on a large Twitter dataset show that there are more meaningful and unique bursty topics in the top-ranked results returned by our model than an LDA baseline and two degenerate variations of our model. We also show some case studies that demonstrate the importance of considering both the temporal information and users' personal interests for bursty topic detection from microblogs.

## 1 Introduction

With the fast growth of Web 2.0, a vast amount of user-generated content has accumulated on the social Web. In particular, microblogging sites such as Twitter allow users to easily publish short instant posts about any topic to be shared with the

general public. The textual content coupled with the temporal patterns of these microblog posts provides important insight into the general public's interest. A sudden increase of topically similar posts usually indicates a burst of interest in some event that has happened offline (such as a product launch or a natural disaster) or online (such as the spread of a viral video). Finding bursty topics from microblogs therefore can help us identify the most popular events that have drawn the public's attention. In this paper, we study the problem of finding bursty topics from a stream of microblog posts generated by different users. We focus on retrospective detection, where the text stream within a certain period is analyzed in its entirety.

Retrospective bursty event detection from text streams is not new (Kleinberg, 2002; Fung et al., 2005; Wang et al., 2007), but finding bursty topics from microblog streams has not been well studied. In his seminal work, Kleinberg (2002) proposed a state machine to model the arrival times of documents in a stream in order to identify bursts. This model has been widely used. However, this model assumes that documents in the stream are all about a given topic. In contrast, discovering interesting topics that have drawn bursts of interest from a stream of topically diverse microblog posts is itself a challenge. To discover topics, we can certainly apply standard topic models such as LDA (Blei et al., 2003), but with standard LDA temporal information is lost during topic discovery. For microblogs, where posts are short and often event-driven, temporal information can sometimes be critical in determining the topic of a post. For example, typically a post containing the

word “jobs” is likely to be about employment, but right after October 5, 2011, a post containing “jobs” is more likely to be related to Steve Jobs’ death. Essentially, we expect that on microblogs, posts published around the same time have a higher probability to belong to the same topic.

To capture this intuition, one solution is to assume that posts published within the same short time window follow the same topic distribution. Wang et al. (2007) proposed a PLSA-based topic model that exploits this idea to find correlated bursty patterns across multiple text streams. However, their model is not immediately applicable for our problem. First, their model assumes multiple text streams where word distributions for the same topic are different on different streams. More importantly, their model was applied to news articles and scientific publications, where most documents follow the global topical trends. On microblogs, besides talking about global popular events, users also often talk about their daily lives and personal interests. In order to detect global bursty events from microblog posts, it is important to filter out these “personal” posts.

In this paper, we propose a topic model designed for finding bursty topics from microblogs. Our model is based on the following two assumptions: (1) If a post is about a global event, it is likely to follow a global topic distribution that is time-dependent. (2) If a post is about a personal topic, it is likely to follow a personal topic distribution that is more or less stable over time. Separation of “global” and “personal” posts is done in an unsupervised manner through hidden variables. Finally, we apply a state machine to detect bursts from the discovered topics.

We evaluate our model on a large Twitter dataset. We find that compared with bursty topics discovered by standard LDA and by two degenerate variations of our model, bursty topics discovered by our model are more accurate and less redundant within the top-ranked results. We also use some example bursty topics to explain the advantages of our model.

## 2 Related Work

To find bursty patterns from data streams, Kleinberg (2002) proposed a state machine to model the arrival times of documents in a stream. Different states generate time gaps according to exponential density

functions with different expected values, and bursty intervals can be discovered from the underlying state sequence. A similar approach by Ihler et al. (2006) models a sequence of count data using Poisson distributions. To apply these methods to find bursty topics, the data stream used must represent a single topic.

Fung et al. (2005) proposed a method that identifies both topics and bursts from document streams. The method first finds individual words that have bursty patterns. It then finds groups of words that tend to share bursty periods and co-occur in the same documents to form topics. Weng and Lee (2011) proposed a similar method that first characterizes the temporal patterns of individual words using wavelets and then groups words into topics. A major problem with these methods is that the word clustering step can be expensive when the number of bursty words is large. We find that the method by Fung et al. (2005) cannot be applied to our dataset because their word clustering algorithm does not scale up. Weng and Lee (2011) applied word clustering to only the top bursty words within a single day, and subsequently their topics mostly consist of two or three words. In contrast, our method is scalable and each detected bursty topic is directly associated with a word distribution and a set of tweets (see Table 3), which makes it easier to interpret the topic.

Topic models provide a principled and elegant way to discover hidden topics from large document collections. Standard topic models do not consider temporal information. A number of temporal topic models have been proposed to consider topic changes over time. Some of these models focus on the change of topic composition, i.e. word distributions, which is not relevant to bursty topic detection (Blei and Lafferty, 2006; Nallapati et al., 2007; Wang et al., 2008). Some other work looks at the temporal evolution of topics, but the focus is not on bursty patterns (Wang and McCallum, 2006; Ahmed and Xing, 2008; Masada et al., 2009; Ahmed and Xing, 2010; Hong et al., 2011).

The model proposed by Wang et al. (2007) is the most relevant to ours. But as we have pointed out in Section 1, they do not need to handle the separation of “personal” documents from event-driven documents. As we will show later in our experiments, for microblogs it is critical to model users’

personal interests in addition to global topical trends.

To capture users' interests, Rosen-Zvi et al. (2004) expand topic distributions from document-level to user-level in order to capture users' specific interests. But on microblogs, posts are short and noisy, so Zhao et al. (2011) further assume that each post is assigned a single topic and some words can be background words. However, these studies do not aim to detect bursty patterns. Our work is novel in that it combines users' interests and temporal information to detect bursty topics.

### 3 Method

#### 3.1 Preliminaries

We first introduce the notation used in this paper and formally formulate our problem. We assume that we have a stream of  $D$  microblog posts, denoted as  $d_1, d_2, \dots, d_D$ . Each post  $d_i$  is generated by a user  $u_i$ , where  $u_i$  is an index between 1 and  $U$ , and  $U$  is the total number of users. Each  $d_i$  is also associated with a discrete timestamp  $t_i$ , where  $t_i$  is an index between 1 and  $T$ , and  $T$  is the total number of time points we consider. Each  $d_i$  contains a bag of words, denoted as  $\{w_{i,1}, w_{i,2}, \dots, w_{i,N_i}\}$ , where  $w_{i,j}$  is an index between 1 and  $V$ , and  $V$  is the vocabulary size.  $N_i$  is the number of words in  $d_i$ .

We define a bursty topic  $b$  as a word distribution coupled with a bursty interval, denoted as  $(\phi^b, t_s^b, t_e^b)$ , where  $\phi^b$  is a multinomial distribution over the vocabulary, and  $t_s^b$  and  $t_e^b$  ( $1 \leq t_s^b \leq t_e^b \leq T$ ) are the start and the end timestamps of the bursty interval, respectively. Our task is to find meaningful bursty topics from the input text stream.

Our method consists of a topic discovery step and a burst detection step. At the topic discovery step, we propose a topic model that considers both users' topical interests and the global topic trends. Burst detection is done through a standard state machine method.

#### 3.2 Our Topic Model

We assume that there are  $C$  (latent) topics in the text stream, where each topic  $c$  has a word distribution  $\phi^c$ . Note that not every topic has a bursty interval. On the other hand, a topic may have multiple bursty intervals and hence leads to multiple bursty topics.

We also assume a background word distribution  $\phi^B$  that captures common words. All posts are assumed to be generated from some mixture of these  $C + 1$  underlying topics.

In standard LDA, a document contains a mixture of topics, represented by a topic distribution, and each word has a hidden topic label. While this is a reasonable assumption for long documents, for short microblog posts, a single post is most likely to be about a single topic. We therefore associate a single hidden variable with each post to indicate its topic. Similar idea of assigning a single topic to a short sequence of words has been used before (Gruber et al., 2007; Zhao et al., 2011). As we will see very soon, this treatment also allows us to model topic distributions at time window level and user level.

As we have discussed in Section 1, an important observation we have is that when everything else is equal, a pair of posts published around the same time is more likely to be about the same topic than a random pair of posts. To model this observation, we assume that there is a global topic distribution  $\theta^t$  for each time point  $t$ . Presumably  $\theta^t$  has a high probability for a topic that is popular in the microblogosphere at time  $t$ .

Unlike news articles from traditional media, which are mostly about current affairs, an important property of microblog posts is that many posts are about users' personal encounters and interests rather than global events. Since our focus is to find popular global events, we need to separate out these "personal" posts. To do this, an intuitive idea is to compare a post with its publisher's general topical interests observed over a long time. If a post does not match the user's long term interests, it is more likely related to a global event. We therefore introduce a time-independent topic distribution  $\eta^u$  for each user to capture her long term topical interests.

We assume the following generation process for all the posts in the stream. When user  $u$  publishes a post at time point  $t$ , she first decides whether to write about a global trendy topic or a personal topic. If she chooses the former, she then selects a topic according to  $\theta^t$ . Otherwise, she selects a topic according to her own topic distribution  $\eta^u$ . With the chosen topic, words in the post are generated from the word distribution for that topic or from the background word distribution that captures white noise.

- 
1. Draw  $\phi^B \sim \text{Dirichlet}(\beta), \pi \sim \text{Beta}(\gamma), \rho \sim \text{Beta}(\lambda)$
  2. For each time point  $t = 1, \dots, T$ 
    - (a) draw  $\theta^t \sim \text{Dirichlet}(\alpha)$
  3. For each user  $u = 1, \dots, U$ 
    - (a) draw  $\eta^u \sim \text{Dirichlet}(\alpha)$
  4. For each topic  $c = 1, \dots, C$ ,
    - (a) draw  $\phi^c \sim \text{Dirichlet}(\beta)$
  5. For each post  $i = 1, \dots, D$ ,
    - (a) draw  $y_i \sim \text{Bernoulli}(\pi)$
    - (b) draw  $z_i \sim \text{Multinomial}(\eta^{u_i})$  if  $y_i = 0$  or  $z_i \sim \text{Multinomial}(\theta^{t_i})$  if  $y_i = 1$
    - (c) for each word  $j = 1, \dots, N_i$ 
      - i. draw  $x_{i,j} \sim \text{Bernoulli}(\rho)$
      - ii. draw  $w_{i,j} \sim \text{Multinomial}(\phi^B)$  if  $x_{i,j} = 0$  or  $w_{i,j} \sim \text{Multinomial}(\phi^{z_i})$  if  $x_{i,j} = 1$
- 

Figure 2: The generation process for all posts.

We use  $\pi$  to denote the probability of choosing to talk about a global topic rather than a personal topic.

Formally, the generation process is summarized in Figure 2. The model is also depicted in Figure 1(a).

There are two degenerate variations of our model that we also consider in our experiments. The first one is depicted in Figure 1(b). In this model, we only consider the time-dependent topic distributions that capture the global topical trends. This model can be seen as a direct application of the model by Wang et al. (2007). The second one is depicted in Figure 1(c). In this model, we only consider the users' personal interests but not the global topical trends, and therefore temporal information is not used. We refer to our complete model as *TimeUserLDA*, the model in Figure 1(b) as *TimeLDA* and the model in Figure 1(c) as *UserLDA*. We also consider a standard LDA model in our experiments, where each word is associated with a hidden topic.

## Learning

We use collapsed Gibbs sampling to obtain samples of the hidden variable assignment and to estimate the model parameters from these samples. Due to space limit, we only show the derived Gibbs sampling formulas as follows.

First, for the  $i$ -th post, we know its publisher  $u_i$  and timestamp  $t_i$ . We can jointly sample  $y_i$  and  $z_i$  based on the values of all other hidden variables. Let us use  $\mathbf{y}$  to denote the set of all hidden variables  $y$  and  $\mathbf{y}_{-i}$  to denote all  $y$  except  $y_i$ . We use similar symbols for other variables. We then have

$$p(y_i = p, z_i = c | \mathbf{z}_{-i}, \mathbf{y}_{-i}, \mathbf{x}, \mathbf{w}) \propto \frac{M_{(\cdot)}^{\pi} + \gamma}{M_{(\cdot)}^{\pi} + 2\gamma} \cdot \frac{M_{(c)}^l + \alpha}{M_{(\cdot)}^l + C\alpha} \cdot \frac{\prod_{v=1}^V \prod_{k=0}^{E_{(v)}-1} (M_{(v)}^c + k + \beta)}{\prod_{k=0}^{E_{(\cdot)}-1} (M_{(\cdot)}^c + k + V\beta)}, \quad (1)$$

where  $l = u_i$  when  $p = 0$  and  $l = t_i$  when  $p = 1$ . Here every  $M$  is a counter.  $M_{(0)}^{\pi}$  is the number of posts generated by personal interests, while  $M_{(1)}^{\pi}$  is the number of posts coming from global topical trends.  $M_{(\cdot)}^{\pi} = M_0^{\pi} + M_1^{\pi}$ .  $M_{(c)}^{u_i}$  is the number of posts by user  $u_i$  and assigned to topic  $c$ , and  $M_{(\cdot)}^{u_i}$  is the total number of posts by  $u_i$ .  $M_{(c)}^{t_i}$  is the number of posts assigned to topic  $c$  at time point  $t_i$ , and  $M_{(\cdot)}^{t_i}$  is the total number of posts at  $t_i$ .  $E_{(v)}$  is the number of times word  $v$  occurs in the  $i$ -th post and is labeled as a topic word, while  $E_{(\cdot)}$  is the total number of topic words in the  $i$ -th post. Here, topic words refer to words whose latent variable  $x$  equals 1.  $M_{(v)}^c$  is the number of times word  $v$  is assigned to topic  $c$ , and  $M_{(\cdot)}^c$  is the total number of words assigned to topic  $c$ . All the counters  $M$  mentioned above are calculated with the  $i$ -th post excluded.

We sample  $x_{i,j}$  for each word  $w_{i,j}$  in the  $i$ -th post using

$$p(x_{i,j} = q | \mathbf{y}, \mathbf{z}, \mathbf{x}_{-\{i,j\}}, \mathbf{w}) \propto \frac{M_{(q)}^{\rho} + \gamma}{M_{(\cdot)}^{\rho} + 2\gamma} \cdot \frac{M_{(w_{i,j})}^l + \beta}{M_{(\cdot)}^l + V\beta}, \quad (2)$$

where  $l = B$  when  $q = 0$  and  $l = z_i$  when  $q = 1$ .  $M_{(0)}^{\rho}$  and  $M_{(1)}^{\rho}$  are counters to record the numbers of words assigned to the background model and any topic, respectively, and  $M_{(\cdot)}^{\rho} = M_{(0)}^{\rho} + M_{(1)}^{\rho}$ .  $M_{(w_{i,j})}^B$  is the number of times word  $w_{i,j}$  occurs as a background word.  $M_{(w_{i,j})}^{z_i}$  counts the number of times word  $w_{i,j}$  is assigned to topic  $z_i$ , and  $M_{(\cdot)}^{z_i}$  is the total number of words assigned to topic  $z_i$ . Again, all counters are calculated with the current word  $w_{i,j}$  excluded.

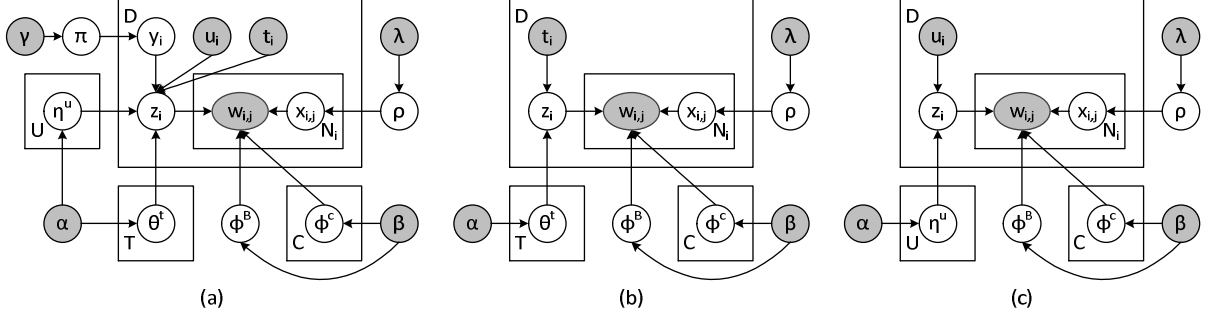


Figure 1: (a) Our topic model for burst detection. (b) A variation of our model where we only consider global topical trends. (c) A variation of our model where we only consider users’ personal topical interests.

### 3.3 Burst Detection

Just like standard LDA, our topic model itself finds a set of topics represented by  $\phi^c$  but does not directly generate bursty topics. To identify bursty topics, we use the following mechanism, which is based on the idea by Kleinberg (2002) and Ihler et al. (2006). In our experiments, when we compare different models, we also use the same burst detection mechanism for other models.

We assume that after topic modeling, for each discovered topic  $c$ , we can obtain a series of counts  $(m_1^c, m_2^c, \dots, m_T^c)$  representing the intensity of the topic at different time points. For LDA, these are the numbers of words assigned to topic  $c$ . For TimeUserLDA, these are the numbers of posts which are in topic  $c$  and generated by the global topic distribution  $\theta^{t_i}$ , i.e whose hidden variable  $y_i$  is 1. For other models, these are the numbers of posts in topic  $c$ .

We assume that these counts are generated by two Poisson distributions corresponding to a bursty state and a normal state, respectively. Let  $\mu_0$  denote the expected count for the normal state and  $\mu_1$  for the bursty state. Let  $v_t$  denote the state for time point  $t$ , where  $v_t = 0$  indicates the normal state and  $v_t = 1$  indicates the bursty state. The probability of observing a count of  $m_t^c$  is as follows:

$$p(m_t^c | v_t = l) = \frac{e^{-\mu_l} \mu_l^{m_t^c}}{m_t^c!},$$

where  $l$  is either 0 or 1. The state sequence  $(v_0, v_1, \dots, v_T)$  is a Markov chain with the following transition probabilities:

$$p(v_t = l | v_{t-1} = l) = \sigma_l,$$

Method	P@5	P@10	P@20	P@30
LDA	0.600	0.800	0.750	N/A
TimeLDA	0.800	0.700	0.600	0.633
UserLDA	0.800	0.700	0.850	<b>0.833</b>
TimeUserLDA	<b>1.000</b>	<b>1.000</b>	<b>0.900</b>	0.800

Table 1: Precision at  $K$  for the various models.

Method	P@5	P@10	P@20	P@30
LDA	0.600	0.800	0.700	N/A
TimeLDA	0.400	0.500	0.500	0.567
UserLDA	0.800	0.500	0.500	0.600
TimeUserLDA	<b>1.000</b>	<b>0.900</b>	<b>0.850</b>	<b>0.767</b>

Table 2: Precision at  $K$  for the various models after we remove redundant bursty topics.

where  $l$  is either 0 or 1.

$\mu_0$  and  $\mu_1$  are topic specific. In our experiments, we set  $\mu_0 = \frac{1}{T} \sum_t m_t^c$ , that is,  $\mu_0$  is the average count over time. We set  $\mu_1 = 3\mu_0$ . For transition probabilities, we empirically set  $\sigma_0 = 0.9$  and  $\sigma_1 = 0.6$  for all topics.

We can use dynamic programming to uncover the underlying state sequence for a series of counts. Finally, a burst is marked by a consecutive subsequence of bursty states.

## 4 Experiments

### 4.1 Data Set

We use a Twitter data set to evaluate our models. The original data set contains 151,055 Twitter users based in Singapore and their tweets. These Twitter users were obtained by starting from a set of seed Singapore users who are active online and tracing



Bursty Period	Top Words	Example Tweets	Label
Nov 29	vote, big, awards, bang, mama, win, 2ne1, award, won	(1) why didnt 2ne1 win this time! (2) 2ne1. you deserved that urgh! (3) watching mama. whoohoo	Mnet Asian Music Awards (MAMA)
Oct 5 ~ Oct 8	steve, jobs, apple, iphone, rip, world, changed, 4s, siri	(1) breaking: apple says steve jobs has passed away! (2) google founders: steve jobs was an inspiration! (3) apple 4 life thankyousteve	Steve Jobs death
Nov 1 ~ Nov 3	reservoir, bedok, adlyn, slap, found, body, mom, singapore, steven	(1) this adelyn totally disgust me. slap her mum? queen of cine? joke please can. (2) she slapped her mum and boasted about it on fb (3) adelyn lives in woodlands , later she slap me how?	girl slapping mom
Nov 5	reservoir, bedok, adlyn, slap, found, body, mom, singapore, steven	(1) bedok = bodies either drowned or killed. (2) another body found, in bedok reservoir? (3) so many bodies found at bedok reservoir. alamak.	suicide near bedok reservoir
Oct 23	man, arsenal, united, liverpool, chelsea, city, goal, game, match	(1) damn you man city! we will get you next time! (2) wtf 90min goal! (3) 6-1 to city. unbelievable.	football game

Table 3: Top-5 bursty topics ranked by TimeUserLDA. The labels are manually given. The 3rd and the 4th bursty topics come from the same topic but have different bursty periods.

Rank	LDA	UserLDA	TimeLDA
1	Steve Jobs' death	MAMA	MAMA
2	MAMA	football game	MAMA
3	N/A	#zamanprimaryschool	MAMA
4	girl slapping mom	N/A	girl slapping mom
5	N/A	iphone 4s	N/A

Table 4: Top-5 bursty topics ranked by other models. N/A indicates a meaningless burst.

their follower/followee links by two hops. Because this data set is huge, we randomly sampled 2892 users from this data set and extracted their tweets between September 1 and November 30, 2011 (91 days in total). We use one day as our time window. Therefore our timestamps range from 1 to 91. We then removed stop words and words containing non-standard characters. Tweets containing less than 3 words were also discarded. After preprocessing, we obtained the final data set with 3,967,927 tweets and 24,280,638 tokens.

## 4.2 Ground Truth Generation

To compare our model with other alternative models, we perform both quantitative and qualitative evaluation. As we have explained in Section 3, each model gives us time series data for a number of topics, and by applying a Poisson-based state machine, we can obtain a set of bursty topics. For each method, we rank the obtained bursty topics by the number

of tweets (or words in the case of the LDA model) assigned to the topics and take the top-30 bursty topics from each model. In the case of the LDA model, only 23 bursty topics were detected. We merged these topics and asked two human judges to judge their quality by assigning a score of either 0 or 1. The judges are graduate students living in Singapore and not involved in this project. The judges were given the bursty period and 100 randomly selected tweets for the given topic within that period for each bursty topic. They can consult external resources to help make judgment. A bursty topic was scored 1 if the 100 tweets coherently describe a bursty event based on the human judge's understanding. The inter-annotator agreement score is 0.649 using Cohen's kappa, showing substantial agreement. For ground truth, we consider a bursty topic to be correct if both human judges have scored it 1. Since some models gave redundant bursty topics, we also asked one of the judges to identify unique bursty

topics from the ground truth bursty topics.

### 4.3 Evaluation

In this section, we show the quantitative evaluation of the four models we consider, namely, LDA, TimeLDA, UserLDA and TimeUserLDA. For each model, we set the number of topics  $C$  to 80,  $\alpha$  to  $\frac{50}{C}$  and  $\beta$  to 0.01 after some preliminary experiments. Each model was run for 500 iterations of Gibbs sampling. We take 40 samples with a gap of 5 iterations in the last 200 iterations to help us assign values to all the hidden variables.

Table 1 shows the comparison between these models in terms of the precision of the top- $K$  results. As we can see, our model outperforms all other models for  $K \leq 20$ . For  $K = 30$ , the UserLDA model performs the best followed by our model.

As we have pointed out, some of the bursty topics are redundant, i.e. they are about the same bursty event. We therefore also calculated precision at  $K$  for unique topics, where for redundant topics the one ranked the highest is scored 1 and the other ones are scored 0. The comparison of the performance is shown in Table 2. As we can see, in this case, our model outperforms other models with all  $K$ . We will further discuss redundant bursty topics in the next section.

### 4.4 Sample Results and Discussions

In this section, we show some sample results from our experiments and discuss some case studies that illustrate the advantages of our model.

First, we show the top-5 bursty topics discovered by the TimeUserLDA model in Table 3. As we can see, all these bursty topics are meaningful. Some of these events are global major events such as Steve Jobs' death, while some others are related to online events such as the scandal of a girl boasting about slapping her mother on Facebook. For comparison, we also show the top-5 bursty topics discovered by other models in Table 4. As we can see, some of them are not meaningful events while some of them are redundant.

Next, we show two case studies to demonstrate the effectiveness of our model.

**Effectiveness of Temporal Models:** Both TimeLDA and TimeUserLDA tend to group posts published on the same day into the same topic. We

find that this can help separate bursty topics from general ones. An example is the topic on the Circle Line. The Circle Line is one of the subway lines of Singapore's mass transit system. There were a few incidents of delays or breakdowns during the period between September and November, 2011. We show the time series data of the topic related to the Circle Line of UserLDA, TimeLDA and TimeUserLDA in Figure 3. As we can see, the UserLDA model detects a much large volume of tweets related to this topic. A close inspection tells us that the topic under UserLDA is actually related to the subway systems in Singapore in general, which include a few other subway lines, and the Circle Line topic is merged with this general topic. On the other hand, TimeLDA and TimeUserLDA are both able to separate the Circle Line topic from the general subway topic because the Circle Line has several bursts. What is shown in Figure 3 for TimeLDA and TimeUserLDA is only the topic on the Circle Line, therefore the volume is much smaller. We can see that TimeLDA and TimeUserLDA show clearer bursty patterns than UserLDA for this topic. The bursts around day 20, day 44 and day 85 are all real events based on our ground truth.

**Effectiveness of User Models:** We have stated that it is important to filter out users' "personal" posts in order to find meaningful global events. We find that our results also support this hypothesis. Let us look at the example of the topic on the Mnet Asian Music Awards, which is a major music award show that is held by Mnet Media annually. In 2011, this event took place in Singapore on November 29. Because Korean pop music is very popular in Singapore, many Twitter users often tweet about Korean pop music bands and singers in general. All our topic models give multiple topics related to Korean pop music, and many of them have a burst on November 29, 2011. Under the TimeLDA and UserLDA models, this leads to several redundant bursty topics for the MAMA event ranked within the top-30. For TimeUserLDA, however, although the MAMA event is also ranked the top, there is no redundant one within the top-30 results. We find that this is because with TimeUserLDA, we can remove tweets that are considered personal and therefore do not contribute to bursty topic ranking. We show the topic intensity of a topic about a Korean pop singer in

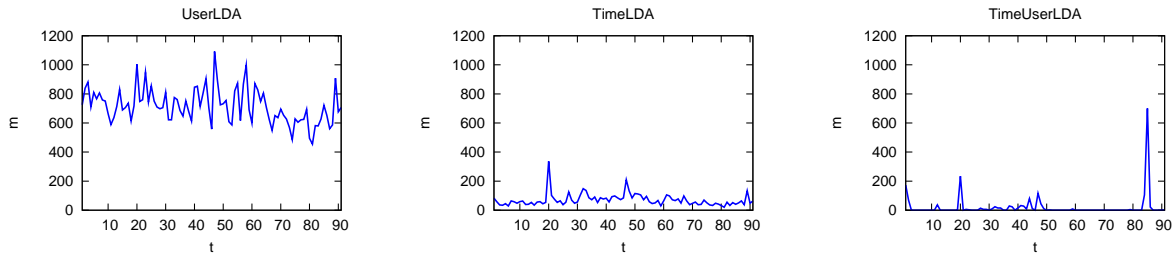


Figure 3: Topic intensity over time for the topic on the Circle Line.

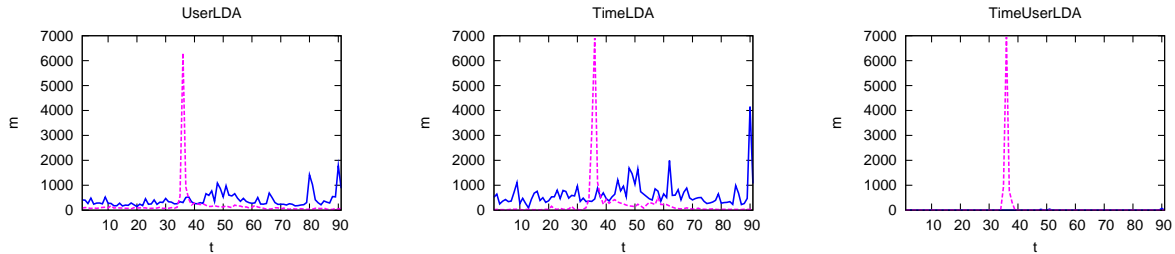


Figure 4: Topic intensity over time for the topic about a Korean pop singer. The dotted curves show the topic on Steve Jobs' death.

Figure 4. For reference, we also show the intensity of the topic on Steve Jobs' death under each model. We can see that because this topic is related to Korean pop music, it has a burst on day 90 (November 29). But if we consider the relative intensity of this burst compared with Steve Jobs' death, under TimeLDA and UserLDA, this topic is still strong but under TimeUserLDA its intensity can almost be ignored. This is why with TimeLDA and UserLDA this topic leads to a redundant burst within the top-30 results but with TimeUserLDA the burst is not ranked high.

## 5 Conclusions

In this paper, we studied the problem of finding bursty topics from the text streams on microblogs. Because existing work on burst detection from text streams may not be suitable for microblogs, we proposed a new topic model that considers both the temporal information of microblog posts and users' personal interests. We then applied a Poisson-based state machine to identify bursty periods from the topics discovered by our model. We compared our model with standard LDA as well as two degenerate variations of our model on a real Twitter dataset. Our quantitative evaluation showed that our

model could more accurately detect unique bursty topics among the top ranked results. We also used two case studies to illustrate the effectiveness of the temporal factor and the user factor of our model.

Our method currently can only detect bursty topics in a retrospective and offline manner. A more interesting and useful task is to detect realtime bursts in an online fashion. This is one of the directions we plan to study in the future. Another limitation of the current method is that the number of topics is predetermined. We also plan to look into methods that allow appearance and disappearance of topics along the timeline, such as the model by Ahmed and Xing (2010).

## Acknowledgments

This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office. We thank the reviewers for their valuable comments.

## References

Amr Ahmed and Eric P. Xing. 2008. Dynamic non-parametric mixture models and the recurrent Chinese

- restaurant process: with applications to evolutionary clustering. In *Proceedings of the SIAM International Conference on Data Mining*, pages 219–230.
- Amr Ahmed and Eric P. Xing. 2010. Timeline: A dynamic hierarchical Dirichlet process model for recovering birth/death and evolution of topics in text stream. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, pages 20–29.
- David M. Blei and John D. Lafferty. 2006. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*.
- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.
- Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Philip S. Yu, and Hongjun Lu. 2005. Parameter free bursty events detection in text streams. In *Proceedings of the 31st International Conference on Very Large Data Bases*, pages 181–192.
- Amit Gruber, Michal Rosen-Zvi, and Yair Weiss. 2007. Hidden topic Markov model. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*.
- Liangjie Hong, Byron Dom, Siva Gurumurthy, and Kostas Tsioutsoulouklis. 2011. A time-dependent topic model for multiple text streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 832–840.
- Alexander Ihler, Jon Hutchins, and Padhraic Smyth. 2006. Adaptive event detection with time-varying poisson processes. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 207–216.
- Jon Kleinberg. 2002. Bursty and hierarchical structure in streams. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 91–101.
- Tomonari Masada, Daiji Fukagawa, Atsuhiko Takasu, Tsuyoshi Hamada, Yuichiro Shibata, and Kiyoshi Oguri. 2009. Dynamic hyperparameter optimization for bayesian topical trend analysis. In *Proceedings of the 18th ACM Conference on Information and knowledge management*, pages 1831–1834.
- Ramesh M. Nallapati, Susan DITmore, John D. Lafferty, and Kin Ung. 2007. Multiscale topic tomography. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 520–529.
- Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494.
- Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-Markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 424–433.
- Xuanhui Wang, ChengXiang Zhai, Xiao Hu, and Richard Sproat. 2007. Mining correlated bursty topic patterns from coordinated text streams. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 784–793.
- Chong Wang, David M. Blei, and David Heckerman. 2008. Continuous time dynamic topic models. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pages 579–586.
- Jianshu Weng and Francis Lee. 2011. Event detection in Twitter. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media*.
- Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. 2011. Comparing twitter and traditional media using topic models. In *Proceedings of the 33rd European conference on Advances in information retrieval*, pages 338–349.

# Spice it Up? Mining Refinements to Online Instructions from User Generated Content

**Gregory Druck**  
Yahoo! Research  
gdruck@gmail.com

**Bo Pang**  
Yahoo! Research  
bopang42@gmail.com

## Abstract

There are a growing number of popular web sites where users submit and review instructions for completing tasks as varied as building a table and baking a pie. In addition to providing their subjective evaluation, reviewers often provide *actionable* refinements. These refinements clarify, correct, improve, or provide alternatives to the original instructions. However, identifying and reading all relevant reviews is a daunting task for a user. In this paper, we propose a generative model that jointly identifies user-proposed refinements in instruction reviews at multiple granularities, and aligns them to the appropriate steps in the original instructions. Labeled data is not readily available for these tasks, so we focus on the unsupervised setting. In experiments in the recipe domain, our model provides 90.1%  $F_1$  for predicting refinements at the review level, and 77.0%  $F_1$  for predicting refinement segments within reviews.

## 1 Introduction

People turn to the web to seek advice on a wide variety of subjects. An analysis of web search queries posed as questions revealed that “how to” questions are the most popular (Pang and Kumar, 2011). People consult online resources to answer technical questions like “how to put music on my ipod,” and to find instructions for tasks like tying a tie and cooking Thanksgiving dinner. Not surprisingly, there are many Web sites dedicated to providing instructions. For instance, on the popular DIY site *instructables.com* (“share what you

make”), users post instructions for making a wide variety of objects ranging from bed frames to “The Stirling Engine, absorb energy from candles, coffee, and more!”<sup>1</sup> There are also sites like *allrecipes.com* that are dedicated to a specific domain. On these community-based instruction sites, instructions are posted and reviewed by users. For instance, the aforementioned “Stirling engine” has received over 350 reviews on *instructables.com*.

While user-generated instructions greatly increase the variety of instructions available online, they are not necessarily foolproof, or appropriate for all users. For instance, in the case of recipes, a user missing a certain ingredient at home might wonder whether it can be safely omitted; a user who wants to get a slightly different flavor might want to find out what substitutions can be used to achieve that effect. Reviews posted by other users provide a great resource for mining such information. In recipe reviews, users often offer their customized version of the recipe by describing changes they made: e.g., “I halved the salt” or “I used honey instead of sugar.” In addition, they may clarify portions of the instructions that are too concise for a novice to follow, or describe changes to the cooking method that result in a better dish. We refer to such *actionable* information as a *refinement*.

Refinements can be quite prevalent in instruction reviews. In a random sample of recipe reviews from *allrecipes.com*, we found that 57.8% contain refinements of the original recipe. However, sifting through all reviews for refinements is a daunting

---

<sup>1</sup><http://www.instructables.com/id/The-Sterling-Engine-absorb-energy-from-candles-c>

task for a user. Instead, we would like to automatically identify refinements in reviews, summarize them, and either create an annotated version of the instructions that reflects the collective experience of the community, or, more ambitiously, revise the instructions directly.

In this paper, we take first steps toward these goals by addressing the following tasks: (1) identifying reviews that contain refinements, (2) identifying text segments within reviews that describe refinements, and (3) aligning these refinement segments to steps in the instructions being reviewed (Figure 1 provides an example). Solving these tasks provides a foundation for downstream summarization and semantic analysis, and also suggests intermediate applications. For example, we can use review classification to filter or rank reviews as they are presented to future users, since reviews that contain refinements are more informative than a review which only says “Great recipe, thanks for posting!”

To the best of our knowledge, no previous work has explored this aspect of user-generated text. While review mining has been studied extensively, we differ from previous work in that instead of focusing on *evaluative* information, we focus *actionable* information in the reviews. (See Section 2 for a more detailed discussion.)

There is no existing labeled data for the tasks of interest, and we would like the methods we develop to be easily applied in multiple domains. Motivated by this, we propose a generative model for solving these tasks jointly without labeled data. Interestingly, we find that jointly modeling refinements at both the review and segment level is beneficial. We created a new recipe data set, and manually labeled a random sample to evaluate our model and several baselines. We obtain 90.1%  $F_1$  for predicting refinements at the review level, and 77.0%  $F_1$  for predicting refinement segments within reviews.

## 2 Related Work

At first glance, the task of identifying refinements appears similar to subjectivity detection (see (Pang and Lee, 2008) for a survey). However, note that an objective sentence is not necessarily a refinement: e.g., “I took the cake to work”; and a subjective sentence can still contain a refinement: e.g., “I reduced

the sugar and it came out perfectly.”

Our end goal is similar to review summarization. However, previous work on review summarization (Hu and Liu, 2004; Popescu and Etzioni, 2005; Titov and McDonald, 2008) in product or service domains focused on summarizing evaluative information — more specifically, identifying ratable aspects (e.g., “food” and “service” for restaurants) and summarizing the overall sentiment polarity for each aspect. In contrast, we are interested in extracting a subset of the non-evaluative information. Rather than ratable aspects that are common across the entire domain (e.g., “ingredient”, “cooking method”), we are interested in actionable information that is related and *specific* to the subject of the review.

Note that while our end goal is to summarize objective information, it is still very different from standard multi-document summarization (Radev et al., 2002) of news articles. Apart from differences in the quantity and the nature of the input, we aim to summarize a distribution over what should or can be changed, rather than produce a consensus using different accounts of an event. In terms of modeling approaches, in the context of extractive summarization, Barzilay and Lee (2004) model content structure (i.e., the order in which topics appear) in documents. We also model document structure, but we do so to help identify refinement segments.

We share with previous work on predicting review quality or helpfulness an interest in identifying “informative” text. Early work tried to exploit the intuition that a helpful review is one that comments on product details. However, incorporating product-aspect-mention count (Kim et al., 2006) or similarity between the review and product specification (Zhang and Varadarajan, 2006) as features did not seem to improve the performance when the task was predicting the percentage of helpfulness votes. Instead of using the helpfulness votes, Liu et al. (2007) manually annotated reviews with quality judgements, where a *best* review was defined as one that contains complete and detailed comments. Our notion of informativeness differs from previous work. We do not seek reviews that contain detailed *evaluative* information; instead, we seek reviews that contain detailed *actionable* information. Furthermore, we are not expecting any single review to be comprehensive; rather, we seek to extract a

collection of refinements representing the collective wisdom of the community.

To the best of our knowledge, there is little previous work on mining user-generated data for actionable information. However, there has been increasing interest in language grounding. In particular, recent work has studied learning to act in an external environment by following textual instructions (Branavan et al., 2009, 2010, 2011; Vogel and Jurafsky, 2010). This line of research is complementary to our work. While we do not utilize extensive linguistic knowledge to analyze actionable information, we view this as an interesting future direction.

We propose a generative model that makes predictions at both the review and review segment level. Recent work uses a discriminative model with a similar structure to perform sentence-level sentiment analysis with review-level supervision (Täckström and McDonald, 2011). However, sentiment polarity labels at the review level are easily obtained. In contrast, refinement labels are not naturally available, motivating the use of unsupervised learning. Note that the model of Täckström and McDonald (2011) cannot be used in a fully unsupervised setting.

### 3 Refinements

In this section, we define refinements more precisely. We use recipes as our running example, but our problem formulation and models are not specific to this domain.

A *refinement* is a piece of text containing actionable information that is not entailed by the original instructions, but can be used to modify or expand the original instructions. A refinement could propose an alternative method or an improvement (e.g., “I replaced half of the shortening with butter”, “Let the shrimp sit in 1/2 marinade for 3 hours”), as well as provide clarification (“definitely use THIN cut pork chops, otherwise your panko will burn before your chops are cooked”).

Furthermore, we distinguish between a *verified* refinement (what the user actually did) and a *hypothetical* refinement (“next time I think I will try evaporated milk”). In domains similar to recipes, where instructions may be carried out repeatedly, there exist refinements in both forms. Since instructions should, in principle, contain information that

has been well tested, in this work, we consider only the former as our target class. In a small percentage of reviews we observed “failed attempts” where a user did not follow a certain step and regretted the diversion. In this work, we do not consider them to be refinements. We refer to text that does not contain refinements as *background*.

Finally, we note that the presence of a past tense verb does not imply a refinement (e.g., “Everyone *loved* this dish”, “I *got* many compliments”). In fact, not all text segments that describe an action are refinements (e.g., “I *took* the cake to work”, “I *followed* the instructions to a T”).

## 4 Models

In this section we describe our models. To identify refinements without labeled data, we propose a generative model of reviews (or more generally documents) with latent variables. We assume that each review  $x$  is divided into *segments*,  $x = (x_1, \dots, x_T)$ . Each segment is a sub-sentence-level text span. We assume that the segmentation is observed, and hence it is not modeled. The segmentation procedure we use is described in Section 5.1.

While we focus on the unsupervised setting, note that the model can also be used in a semi-supervised setting. In particular, coarse (review-level) labels can be used to guide the induction of fine-grained latent structure (segment labels, alignments).

### 4.1 Identifying Refinements

We start by directly modeling refinements at the segment level. Our first intuition is that refinement and background segments can often be identified by lexical differences. Based on this intuition, we can ignore document structure and generate the segments with a segment-level mixture of multinomials (S-Mix). In general we could use  $n$  multinomials to represent refinements and  $m$  multinomials to represent background text, but in this paper we simply use  $n = m = 1$ . Therefore, unsupervised learning in S-Mix can be viewed as clustering the segments with two latent states. As is standard practice in unsupervised learning, we subsequently map these latent states onto the labels of interest:  $r$  and  $b$ , for refinement and background, respectively. Note, however, that this model ignores potential sequential depen-

dencies among segments. A segment following a refinement segment in a review may be more likely to be a refinement than background, for example.

To incorporate this intuition, we could instead generate reviews with a HMM (Rabiner, 1989) over segments (S-HMM) with two latent states. Let  $z_i$  be the latent label variable for the  $i$ th segment. The joint probability of a review and segment labeling is

$$p(\mathbf{x}, \mathbf{z}; \theta) = \prod_{j=1}^T p(z_j | z_{j-1}; \theta) p(\mathbf{x}_j | z_j; \theta), \quad (1)$$

where  $p(z_j | z_{j-1}; \theta)$  are multinomial transition distributions, allowing the model to learn that  $p(z_j = r | z_{j-1} = r; \theta) > p(z_j = b | z_{j-1} = r; \theta)$  as motivated above, and  $p(\mathbf{x}_j | z_j; \theta)$  are multinomial emission distributions. Note that all words in a segment are generated independently conditioned on  $z_j$ .

While S-HMM models sequential dependencies, note that it imposes the same transition probabilities on each review. In a manually labeled random sample of recipe reviews, we find that refinement segments tend to be clustered together in certain reviews (“bursty”), rather than uniformly distributed across all reviews. Specifically, while we estimate that 23% of all segments are refinements, 42% of reviews do not contain any refinements. In reviews that contain a refinement, 34% of segments are refinements. S-HMM cannot model this phenomenon.

Consequently, we extend S-HMM to include a latent label variable  $y$  for each review that takes values *yes* (contains refinement) and *no* (does not contain refinement). The extended model is a mixture of HMMs (RS-MixHMM) where  $y$  is the mixture component.

$$p(\mathbf{x}, y, \mathbf{z}; \theta) = p(y; \theta) p(\mathbf{x}, \mathbf{z} | y; \theta) \quad (2)$$

The two HMMs  $p(\mathbf{x}, \mathbf{z} | y = \textit{yes}; \theta)$  and  $p(\mathbf{x}, \mathbf{z} | y = \textit{no}; \theta)$  can learn different transition multinomials and consequently different distributions over  $\mathbf{z}$  for different  $y$ . On the other hand, we do not believe the textual content of the background segments in a  $y = \textit{yes}$  review should be different from those in a  $y = \textit{no}$  review. Thus, the emission distributions are shared between the two HMMs,  $p(\mathbf{x}_j | z_j, y; \theta) = p(\mathbf{x}_j | z_j; \theta)$ .

Note that the definition of  $y$  imposes additional constraints on RS-MixHMM: 1) reviews with  $y = \textit{no}$

cannot contain refinement segments, and 2) reviews with  $y = \textit{yes}$  must contain at least one refinement segment. We enforce constraint (1) by disallowing refinement segments  $z_j = r$  when  $y = \textit{no}$ :  $p(z_j = r | z_{j-1}, y = \textit{no}; \theta) = 0$ . Therefore, with one background label, only the all background label sequence has non-zero probability when  $y = \textit{no}$ . Enforcing constraint (2) is more challenging, as the  $y = \textit{yes}$  HMM must assign zero probability when all segments are background, but permit background segments when refinement segments are present.

To enforce constraint (2), we “rewire” the HMM structure for  $y = \textit{yes}$  so that a path that does not go through the refinement state  $r$  is impossible. We first expand the state representation by replacing  $b$  with two states that encode whether or not the first  $r$  has been encountered yet:  $b_{\textit{not-yet}}$  encodes that all previous states in the path have also been background;  $b_{ok}$  encodes that at least one refinement state has been encountered<sup>2</sup>. We prohibit paths from ending with  $b_{\textit{not-yet}}$  by augmenting RS-MixHMM with a special final state  $f$ , and fixing  $p(z_{T+1} = f | z_T = b_{\textit{not-yet}}, y = \textit{yes}; \theta) = 0$ . Furthermore, to enforce the correct semantics of each state, paths cannot start with  $b_{ok}$ ,  $p(z_1 = b_{ok} | y = \textit{yes}; \theta) = 0$ , and transitions from  $b_{\textit{not-yet}}$  to  $b_{ok}$ ,  $b_{ok}$  to  $b_{\textit{not-yet}}$ , and  $r$  to  $b_{\textit{not-yet}}$  are prohibited.

Note that RS-MixHMM also generalizes to the case where there are multiple refinement ( $n > 1$ ) and background ( $m > 1$ ) labels. Let  $\mathcal{Z}_r$  be the set of refinement labels, and  $\mathcal{Z}_b$  be the set of background labels. The transition structure is analogous to the  $n = m = 1$  case, but statements involving  $r$  are applied for each  $z \in \mathcal{Z}_r$ , and statements involving  $b$  are applied for each  $z \in \mathcal{Z}_b$ . For example, the  $y = \textit{yes}$  HMM contains  $2|\mathcal{Z}_b|$  background states.

In summary, the generative process of RS-MixHMM involves first selecting whether the review will contain a refinement. If the answer is *yes*, a sequence of background segments and at least one refinement segment are generated using the  $y = \textit{yes}$  HMM. If the answer is *no*, only background segments are generated. Interestingly, by enforcing constraints (1) and (2), we break the label symmetry that necessitates mapping latent states onto labels

<sup>2</sup>In this paper, the two background states share emission multinomials,  $p(\mathbf{x}_j | z_j = b_{\textit{not-yet}}; \theta) = p(\mathbf{x}_j | z_j = b_{ok}; \theta)$ , though this is not required.



when using S-Mix and S-HMM. Indeed, in the experiments we present in Section 5.3, mapping is not necessary for RS-MixHMM.

Note that the relationship between document-level labels and segment-level labels that we model is related to the *multiple-instance* setting (Dietterich et al., 1997) in the machine learning literature. In multiple-instance learning (MIL), rather than having explicit labels at the instance (e.g., segment) level, labels are given for bags of instances (e.g., documents). In the binary case, a bag is *negative* only if all of its instances are *negative*. While we share this problem formulation, work on MIL has mostly focussed on supervised learning settings, and thus it is not directly applicable to our unsupervised setting. Foulds and Smyth (2011) propose a generative model for MIL in which the generation of the bag label  $y$  is conditioned on the instance labels  $\mathbf{z}$ . As a result of this setup, their model reduces to our S-Mix baseline in a fully unsupervised setting.

Finally, although we motivated including the review-level latent variable  $y$  as a way to improve segment-level prediction of  $\mathbf{z}$ , note that predictions of  $y$  are useful in and of themselves. They provide some notion of review *usefulness* and can be used to filter reviews for search and browsing. They additionally give us a way to measure whether a set of instructions is often modified or performed as specified. Finally, if we want to provide supervision, it is much easier to annotate whether a review contains a refinement than to annotate each segment.

## 4.2 Alignment with the Instructions

In addition to the review  $\mathbf{x}$ , we also observe the set of instructions  $\mathbf{s}$  being discussed. Often a review will reference specific parts of the instructions. We assume that each set of instructions is segmented into *steps*,  $\mathbf{s} = (s_1, \dots, s_S)$ . We augment our model with latent *alignment* variables  $\mathbf{a} = (a_1, \dots, a_T)$ , where  $a_j = \ell$  denotes that the  $j$ th review segment is referring to the  $\ell$ th step of  $\mathbf{s}$ . We also define a special NULL instruction step. An alignment to NULL signifies that the segment does not refer to a specific instruction step. Note that this encoding assumes that each review segment refers to at most one instruction step. Alignment predictions could facilitate further analysis of how refinements affect the instructions, as well as aid in summarization and visualization of

refinements.

The joint probability under the augmented model, which we refer to as RSA-MixHMM, is

$$p(\mathbf{a}, \mathbf{x}, y, \mathbf{z} | \mathbf{s}; \boldsymbol{\theta}) = p(y; \boldsymbol{\theta}) p(\mathbf{a}, \mathbf{x}, \mathbf{z} | y, \mathbf{s}; \boldsymbol{\theta}) \quad (3)$$

$$p(\mathbf{a}, \mathbf{x}, \mathbf{z} | y, \mathbf{s}; \boldsymbol{\theta}) = \prod_{j=1}^T p(a_j, z_j | a_{j-1}, z_{j-1}, y, \mathbf{s}; \boldsymbol{\theta})$$

$$\times p(\mathbf{x}_j | a_j, z_j, \mathbf{s}; \boldsymbol{\theta}).$$

Note that the instructions  $\mathbf{s}$  are assumed to be observed and hence are not generated by the model. RSA-MixHMM can be viewed as a mixture of HMMs where each state encodes both a segment label  $z_j$  and an alignment variable  $a_j$ . Encoding an alignment problem as a sequence labeling problem was first proposed by Vogel et al. (1996). Note that RSA-MixHMM uses a similar expanded state representation and transition structure as RS-MixHMM to encode the semantics of  $y$ .

In our current model, the transition probability decomposes into the product of independent label transition and alignment transition probabilities

$$p(a_j, z_j | a_{j-1}, z_{j-1}, y, \mathbf{s}; \boldsymbol{\theta}) = p(a_j | a_{j-1}, y, \mathbf{s}; \boldsymbol{\theta})$$

$$\times p(z_j | z_{j-1}, y, \mathbf{s}; \boldsymbol{\theta}),$$

and  $p(a_j | a_{j-1}, y, \mathbf{s}; \boldsymbol{\theta}) = p(a_j | y, \mathbf{s}; \boldsymbol{\theta})$  simply encodes the probability that segments align to a (non-NULL) instruction step given  $y$ . This allows the model to learn, for example, that reviews that contain refinements refer to the instructions more often.

Intuitively, a segment and the step it refers to should be lexically similar. Consequently, RSA-MixHMM generates segments using a mixture of the multinomial distribution for the segment label  $z_j$  and the (fixed) multinomial distribution<sup>3</sup> for the step  $s_{a_j}$ . In this paper, we do not model the mixture probability and simply assume that all overlapping words are generated by the instruction step. When  $a_j = \text{NULL}$ , only the segment label multinomial is used. Finally, we disallow an alignment to a non-NULL step if no words overlap:  $p(\mathbf{x}_j | a_j, z_j, \mathbf{s}; \boldsymbol{\theta}) = 0$ .

## 4.3 Inference and Parameter Estimation

Because our model is tree-structured, we can efficiently compute exact marginal distributions

<sup>3</sup>Stopwords are removed from the instruction step.

over latent variables using the *sum-product algorithm* (Koller and Friedman, 2009). Similarly, to find maximum probability assignments, we use the *max-product algorithm*.

At training time we observe a set of reviews and corresponding instructions,  $\mathcal{D} = \{(\mathbf{x}^1, \mathbf{s}^1), \dots, (\mathbf{x}^N, \mathbf{s}^N)\}$ . The other variables,  $y$ ,  $\mathbf{z}$ , and  $\mathbf{a}$ , are latent. For all models, we estimate parameters to maximize the marginal likelihood of the observed reviews. For example, for RSA-MixHMM, we estimate parameters using

$$\arg \max_{\theta} \sum_{i=1}^N \log \sum_{\mathbf{a}, \mathbf{z}, y} p(\mathbf{a}, \mathbf{x}^i, y, \mathbf{z} | \mathbf{s}^i; \theta).$$

This problem cannot be solved analytically, so we use the Expectation Maximization (EM) algorithm.

## 5 Experiments

### 5.1 Data

In this paper, we use recipes and reviews from *allrecipes.com*, an active community where we estimate that the mean number of reviews per recipe is 54.2. We randomly selected 22,437 reviews for our data set. Of these, we randomly selected a subset of 550 reviews and determined whether or not each contains a refinement, using the definition provided in Section 3. In total, 318 of the 550 (57.8%) contain a refinement. We then randomly selected 119 of the 550 and labeled the individual segments. Of the 712 segments in the selected reviews, 165 (23.2%) are refinements and 547 are background.

We now define our review segmentation scheme. Most prior work on modeling latent document substructure uses sentence-level labels (Barzilay and Lee, 2004; Täckström and McDonald, 2011). In the recipe data, we find that sentences often contain both refinement and background segments: “[I used a slow cooker with this recipe and] [it turned out great!]” Additionally, we find that sentences often contain several distinct refinements: “[I set them on top and around the pork and] [tossed in a can of undrained french cut green beans and] [cooked everything on high for about 3 hours].” To make refinements easier to identify, and to facilitate downstream processing, we allow sub-sentence segments.

Our segmentation procedure leverages a phrase structure parser. In this paper we use the Stanford

Parser<sup>4</sup>. Based on a quick manual inspection, domain shift and ungrammatical sentences do cause a significant degradation in parsing accuracy when compared to in-domain data. However, this is acceptable because we only use the parser for segmentation. We first parse the entire review, and subsequently iterate through the tokens, adding a segment break when any of the following conditions is met:

- sentence break (determined by the parser)
- token is a *coordinating conjunction* (CC) with parent other than NP, PP, ADJP
- token is a *comma* (,) with parent other than NP, PP, ADJP
- token is a *colon* (:)

The resulting segmentations are fixed during learning. In future work we could extend our model to additionally identify segment boundaries.

### 5.2 Experimental Setup

We first describe the methods we evaluate. For comparison, we provide results with a baseline that randomly guesses according to the class distribution for each task. We also evaluate a **Review-level** model:

- **R-Mix**: A review-level mixture of multinomials with two latent states.

Note that this is similar to clustering at the review level, except that class priors are estimated. R-Mix does not provide segment labels, though they can be obtained by labeling all segments with the review label.

We also evaluate the two **Segment-level** models described in Section 4.1 (with two latent states):

- **S-Mix**: A segment-level mixture model.
- **S-HMM**: A segment-level HMM (Eq. 1).

These models do not provide review labels. To obtain them, we assign  $y = \text{yes}$  if any segment is labeled as a refinement, and  $y = \text{no}$  otherwise.

Finally, we evaluate three versions of our model (**Review + Segment** and **Review + Segment +**

<sup>4</sup><http://nlp.stanford.edu/software/lex-parser.shtml>

Alignment) with one refinement segment label and one background segment label<sup>5</sup>:

- **RS-MixHMM**: A mixture of HMMs (Eq. 2) with constraints (1) and (2) (see Section 4).
- **RS-MixMix**: A variant of RS-MixHMM without sequential dependencies.
- **RSA-MixHMM**: The full model that also incorporates alignment (Eq. 3).

Segment multinomials are initialized with a small amount of random noise to break the initial symmetry. RSA-MixHMM segment multinomials are instead initialized to the RS-MixHMM solution. We apply add-0.01 smoothing to the emission multinomials and add-1 smoothing to the transition multinomials in the M-step. We estimate parameters with 21,887 unlabeled reviews by running EM until the relative percentage decrease in the marginal likelihood is  $\leq 10^{-4}$  (typically 10-20 iterations).

The models are evaluated on refinement  $F_1$  and accuracy for both review and segment predictions using the annotated data described in Section 5.1. For R-Mix and the segment (S-) models, we select the 1:1 mapping of latent states to labels that maximizes  $F_1$ . For RSA-MixHMM and the RS- models this was not necessary (see Section 4.1).

### 5.3 Results

Table 1 displays the results. R-Mix fails to accurately distinguish refinement and background reviews. The words that best discriminate the two discovered review classes are “savory ingredients” (*chicken, pepper, meat, garlic, soup*) and “baking/dessert ingredients” (*chocolate, cake, pie, these, flour*). In other words, reviews naturally cluster by topics rather than whether they contain refinements.

The segment models (S-) substantially outperform R-Mix on all metrics, demonstrating the benefit of segment-level modeling and our segmentation scheme. However, S-HMM fails to model the “burstiness” of refinement segments (see Section 4.1). It predicts that 76.2% of reviews contain refinements, and additionally that 40.9% of segments contain refinements, whereas the true values

<sup>5</sup>Attempts at modeling refinement and background subtypes by increasing the number of latent states failed to substantially improve the results.

are 57.8% and 23.2%, respectively. As a result, these models provide high recall but low precision.

In comparison, our models, which model the review labels<sup>6</sup>  $y$ , yield more accurate refinement predictions. They provide statistically significant improvements in review and segment  $F_1$ , as well as accuracy, over the baseline models. RS-MixHMM predicts that 62.9% of reviews contain refinements and 28.2% of segments contain refinements, values that are much closer to the ground truth. The refinement emission distributions for S-HMM and RS-MixHMM are fairly similar, but the probabilities of several key terms like *added, used, and instead* are higher with RS-MixHMM.

The review  $F_1$  results demonstrate that our models are able to very accurately distinguish refinement reviews from background reviews. As motivated in Section 4.1, there are several applications that can benefit from review-level predictions directly. Additionally, note that review labeling is not a trivial task. We trained a supervised logistic regression model with bag-of-words and length features (for both the number of segments and the number of words) using 10-fold cross validation on the labeled dataset. This supervised model yields mean review  $F_1$  of 78.4, 11.7  $F_1$  points below the best unsupervised result<sup>7</sup>.

Augmenting RS-MixMix with sequential dependencies, yielding RS-MixHMM, provides a moderate (though not statistically significant) improvement in segment  $F_1$ . RS-MixHMM learns that refinement reviews typically begin and end with background segments, and that refinement segments tend to appear in succession.

RSA-MixHMM additionally learns that segments in refinement reviews are more likely to align to non-NULL recipe steps. It also encourages the segment multinomials to focus modeling effort on words that appear only in the reviews. As a result, in addition to yielding alignments, RSA-MixHMM provides small improvements over RS-MixHMM (though they are not statistically significant).

<sup>6</sup>We note that enforcing the constraint that a refinement review must contain at least one refinement segment using the method in Section 4.1 provides a statistically significant improvement in review  $F_1$  of 4.0 for RS-MixHMM.

<sup>7</sup>Note that we do not consider this performance to be the upper-bound of supervised approaches; clearly, supervised approaches could benefit from additional labeled data. However, labeled data is relatively expensive to obtain for this task.

Model	review (57.8% refinement)				segment (23.2% refinement)			
	acc	prec	rec	F <sub>1</sub>	acc	prec	rec	F <sub>1</sub>
random baseline	51.2 <sup>†</sup>	57.8	57.8	57.8 <sup>†</sup>	64.4 <sup>†</sup>	23.2	23.2	23.2 <sup>†</sup>
R-Mix	61.5 <sup>†</sup>	69.1	60.4	64.4 <sup>†</sup>	55.8 <sup>†</sup>	27.9	57.6	37.6 <sup>†</sup>
S-Mix	77.5 <sup>†</sup>	72.4	<b>98.7</b>	83.5 <sup>†</sup>	80.6 <sup>†</sup>	54.7	95.2	69.5 <sup>†</sup>
S-HMM	79.8 <sup>†</sup>	74.7	98.4	84.9 <sup>†</sup>	80.3 <sup>†</sup>	54.3	<b>95.8</b>	69.3 <sup>†</sup>
RS-MixMix	87.1	85.4	93.7	89.4	86.4	65.6	86.7	74.7
RS-MixHMM	87.3	85.6	93.7	89.5	87.9	69.7	84.8	76.5
RSA-MixHMM	<b>88.2</b>	<b>87.1</b>	93.4	<b>90.1</b>	<b>88.5</b>	<b>71.7</b>	83.0	<b>77.0</b>

Table 1: Unsupervised experiments comparing models for review and segment refinement identification on the recipe data. Bold indicates the best result, and a † next to an accuracy or F<sub>1</sub> value indicates that the improvements obtained by RS-MixMix, RS-MixHMM, and RSA-MixHMM are significant ( $p = 0.05$  according to a bootstrap test).

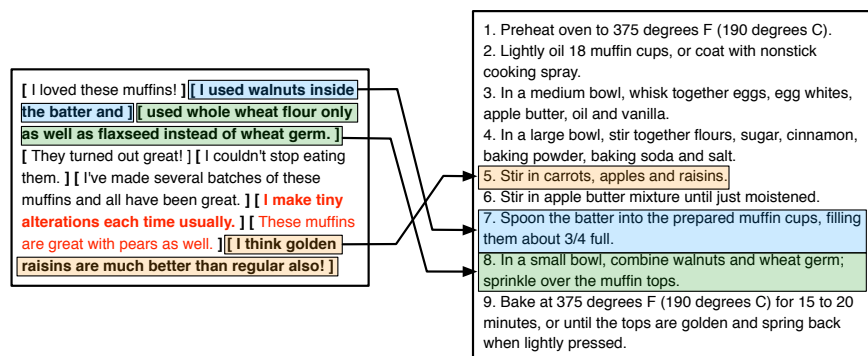


Figure 1: Example output (best viewed in color). Bold segments in the review (left) are those predicted to be refinements. Red indicates an incorrect segment label, according to our gold labels. Alignments to recipe steps (right) are indicated with colors and arrows. Segments without colors and arrows align to the NULL recipe step (see Section 4.2).

We provide an example alignment in Figure 1. Annotating ground truth alignments is challenging and time-consuming due to ambiguity, and we feel that the alignments are best evaluated via a downstream task. Therefore, we leave thorough evaluation of the quality of the alignments to future work.

## 6 Conclusion and Future Work

In this paper, we developed unsupervised methods based on generative models for mining refinements to online instructions from reviews. The proposed models leverage lexical differences in refinement and background segments. By augmenting the base models with additional structure (review labels, alignments), we obtained more accurate predictions.

However, to further improve accuracy, more linguistic knowledge and structure will need to be incorporated. The current models provide many false positives in the more subtle cases, when some words

that typically indicate a refinement are present, but the text does not describe a refinement according to the definition in Section 3. Examples include hypothetical refinements (“next time I will substitute...”) and discussion of the recipe without modification (“I found it strange to... but it worked ...”, “I love balsamic vinegar and herbs”, “they baked up nicely”).

Other future directions include improving the alignment model, for example by allowing words in the instruction step to be “translated” into words in the review segment. Though we focussed on recipes, the models we proposed are general, and could be applied to other domains. We also plan to consider this task in other settings such as online forums, and develop methods for summarizing refinements.

## Acknowledgments

We thank Andrei Broder and the anonymous reviewers for helpful discussions and comments.

## References

- Regina Barzilay and Lillian Lee. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL 2004: Proceedings of the Main Conference*, pages 113–120, 2004.
- S.R.K Branavan, Harr Chen, Luke Zettlemoyer, and Regina Barzilay. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2009.
- S.R.K Branavan, Luke Zettlemoyer, and Regina Barzilay. Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2010.
- S.R.K. Branavan, David Silver, and Regina Barzilay. Learning to win by reading manuals in a monte-carlo framework. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2011.
- Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1 - 2):31 – 71, 1997.
- J. R. Foulds and P. Smyth. Multi-instance mixture models and semi-supervised learning. In *SIAM International Conference on Data Mining*, 2011.
- Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 168–177, 2004.
- Soo-Min Kim, Patrick Pantel, Tim Chklovski, and Marco Pennacchiotti. Automatically assessing review helpfulness. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 423–430, 2006.
- D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- Jingjing Liu, Yunbo Cao, Chin-Yew Lin, Yalou Huang, and Ming Zhou. Low-quality product review detection in opinion summarization. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 334–342, 2007.
- Bo Pang and Ravi Kumar. Search in the lost sense of query: Question formulation in web search queries and its temporal changes. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2011.
- Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.
- Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, 2005.
- Lawrence Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- Dragomir R. Radev, Eduard Hovy, and Kathleen McKeown. Introduction to the special issue on summarization. *Computational Linguistics*, 28(4):399–408, 2002. ISSN 0891-2017.
- Oscar Täckström and Ryan McDonald. Discovering fine-grained sentiment with latent variable structured prediction models. In *Proceedings of the 33rd European conference on Advances in information retrieval, ECIR’11*, pages 368–374, 2011.
- Ivan Titov and Ryan McDonald. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2008.
- Adam Vogel and Daniel Jurafsky. Learning to follow navigational directions. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2010.
- Stephan Vogel, Hermann Ney, and Christoph Tillmann. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics - Volume 2, COLING ’96*, pages 836–841, 1996.
- Zhu Zhang and Balaji Varadarajan. Utility scoring of product reviews. In *Proceedings of the ACM SIGIR Conference on Information and Knowledge Management (CIKM)*, pages 51–57, 2006.

# Sentence Dependency Tagging in Online Question Answering Forums

Zhonghua Qu and Yang Liu  
The University of Texas at Dallas  
{qzh, yangl@hlt.utdallas.edu}

## Abstract

Online forums are becoming a popular resource in the state of the art question answering (QA) systems. Because of its nature as an online community, it contains more updated knowledge than other places. However, going through tedious and redundant posts to look for answers could be very time consuming. Most prior work focused on extracting only question answering sentences from user conversations. In this paper, we introduce the task of sentence dependency tagging. Finding dependency structure can not only help find answer quickly but also allow users to trace back how the answer is concluded through user conversations. We use linear-chain conditional random fields (CRF) for sentence type tagging, and a 2D CRF to label the dependency relation between sentences. Our experimental results show that our proposed approach performs well for sentence dependency tagging. This dependency information can benefit other tasks such as thread ranking and answer summarization in online forums.

## 1 Introduction

Automatic Question Answering (QA) systems rely heavily on good sources of data that contain questions and answers. Question answering forums, such as technical support forums, are places where users find answers through conversations. Because of their nature as online communities, question answering forums provide more updated answers to new problems. For example, when the latest release of Linux has a bug, we can expect to find solutions

in forums first. However, unlike other structured knowledge bases, often it is not straightforward to extract information such as questions and answers in online forums because such information spreads in the conversations among multiple users in a thread.

A lot of previous work has focused on extracting the question and answer sentences from forum threads. However, there is much richer information in forum conversations, and simply knowing a sentence is a question or answer is not enough. For example, in technical support forums, often it takes several iterations of asking and clarifications to describe the question. The same happens to answers. Usually several candidate answers are provided, and not all answers are useful. In this case users' feedback is needed to judge the correctness of answers.

Figure 1 shows an example thread in a technical support forum. Each sentence is labeled with its type (a detailed description of sentence types is provided Table 1). We can see from the example that questions and answers are not expressed in a single sentence or a single post. Only identifying question and answering sentences from the thread is not enough for automatic question answering. For this example, in order to get the complete question, we would need to know that sentence *S3* is a question that inquires for more details about the problem asked earlier, instead of stating its own question. Also, sentence *S5* should not be included in the correct answer since it is not a working solution, which is indicated by a negative feedback in sentence *S6*. The correct solution should be sentence *S7*, because of a user's positive confirmation *S9*. We define that there is a dependency between a pair of sentences if one sentence

**A:** [S1:M-GRET] *Hi everyone.* [S2:P-STAT] *I have recently purchased USB flash and I am having trouble renaming it, please help me.*  
**B:** [S3:A-INQU] *What is the size and brand of this flash?*  
**A:** [S4:Q-CLRF] *It is a 4GB SanDisk flash.*  
**B:** [S5:A-SOLU] *Install gparted, select flash drive and rename.*  
**A:** [S6:M-NEGA] *I got to the Right click on partition and the label option was there but grayed out.*  
**B:** [S7:A-SOLU] *Sorry again, I meant to right click the partition and select Unmount and then select Change name while in gparted.*  
**A:** [S8:C-GRAT] *Thank you so much.* [S9:M-POST] *I now have an "Epic USB" You Rock!*

Figure 1: Example of a Question Answering Thread in Ubuntu Support Forum

exists as a result of another sentence. For example, question context sentences exist because of the question itself; an answering sentence exists because of a question; or a feedback sentence exists because of an answer. The sentence dependency structure of this example dialog is shown in Figure 2.

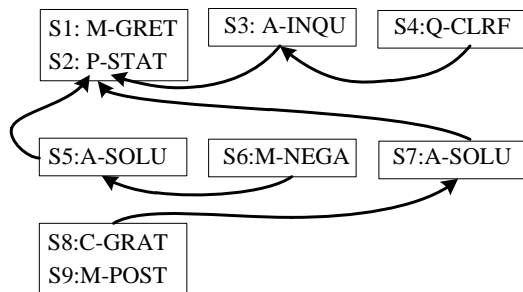


Figure 2: Dependency Structure of the Above Example

This example shows that in order to extract information from QA forums accurately, we need to understand the sentence dependency structure of a QA thread. Towards this goal, in this paper, we define two tasks: labeling the types for sentences, and finding the dependency relations between sentences.

For the first task of sentence type labeling, we define a rich set of categories representing the purpose

of the sentences. We use linear-chain conditional random fields (CRF) to take advantage of many long-distance and non-local features. The second task is to identify relations between sentences. Most previous work only focused on finding the answer-question relationship between sentences. However, other relations can also be useful for information extraction from online threads, such as user’s feedbacks on the answers, problem detail inquiry and question clarifications. In this study, we use two approaches for labeling of dependency relation between sentences. First each sentence is considered as a source, and we run a linear-chain CRF to label whether each of the other sentences is its target. Because multiple runs of separate linear-chain CRFs ignore the dependency between source sentences, the second approach we propose is to use a 2D CRF that models all pair relationships jointly.

The data we used was collected from Ubuntu forum general help section. Our experimental results show that our proposed sentence type tagging method works very well, even for the minority categories, and that using 2D CRF further improves performance over linear-chain CRFs for identifying dependency relation between sentences.

The paper is organized as follows. In the following section, we discuss related work on finding questions and answers in online environment as well as some dialog act tagging techniques. In Section 3, we introduce the use of CRFs for sentence type and dependency tagging. Section 4 describes data collection, annotation, and some analysis. In Section 5, we show that our approach achieves promising results in thread sentence dependency tagging. Finally we conclude the paper and suggest some possible future extensions.

## 2 Related Work

There is a lot of useful knowledge in the user generated content such as forums. This knowledge source could substantially help automatic question answering systems. There has been some previous work focusing on the extraction of question and corresponding answer pairs in online forums. In (Ding et al., 2008), a two-pass approach was used to find relevant solutions for a given question, and a skip-chain CRF was adopted to model long range de-

pendency between sentences. A graph propagation method was used in (Cong et al., 2008) to rank relevant answers to questions. An approach using email structure to detect and summarize question answer pairs was introduced in (Shrestha and Mckeown, 2004). These studies focused primarily on finding questions and answers in an online environment. In this paper, in order to provide a better foundation for question answer detection in online forums, we investigate tagging sentences with a much richer set of categories, as well as identifying their dependency relationships. The sentence types we use are similar to dialog acts (DA), but defined specifically for question answering forums. Work of (Clark and Popescu-Belis, 2004) defined a reusable multi-level tagset that can be mapped from conversational speech corpora such as the ICSI meeting data. However, it is hard to reuse any available corpus or DA tagset because our task is different, and also online forum has a different style from speech data. Automatic DA tagging has been studied a lot previously. For example, in (Stolcke et al., 2000), Hidden Markov Models (HMMs) were used for DA tagging; in (Ji and Bilmes, 2005), different types of graphical models were explored.

Our study is different in several aspects: we are using forum domains, unlike most work of DA tagging on conversational speech; we use CRFs for sentence type tagging; and more importantly, we also propose to use different CRFs for sentence relation detection. Unlike the pair-wise sentence analysis proposed in (Boyer et al., 2009) in which HMM was used to model the dialog structure, our model is more flexible and does not require related sentences to be adjacent.

### 3 Thread Structure Tagging

As described earlier, we decompose the structure analysis of QA threads into two tasks, first determine the sentence type, and then identify related sentences. This section provides details for each task.

#### 3.1 Sentence Type Tagging

In human conversations, especially speech conversations, DAs have been used to represent the purpose or intention of a sentence. Different sets of

DAs have been adopted in various studies, ranging from very coarse categories to fine grained ones. In this study, we define 13 fine grained sentence types (corresponding to 4 coarse categories) tailored to our domain of QA forum threads. Table 1 shows the categories and their description. Some tags such as P-STAT and A-SOLU are more important in that users try to state a problem and provide solutions accordingly. These are the typical ones used in previous work on question answering. Our set includes other useful tags. For example, C-NEGA and C-POSI can evaluate how good an answer is. Even though C-GRAT does not provide any direct feedback on the solutions, existence of such a tag often strongly implies a positive feedback to an answer. These sentence types can be grouped into 4 coarse categories, as shown in Table 1.

Types	Category	Description
Problems	P-STAT	question of problem
	P-CONT	problem context
	P-CLRF	problem clarification
Answers	A-SOLU	solution sentence
	A-EXPL	explanation on solutions
	A-INQU	inquire problem details
Confirm.	C-GRAT	gratitude
	C-NEGA	negative feedback
	C-POSI	positive feedback
Misc.	M-QCOM	question comment
	M-ACOM	comment on the answer
	M-GRET	greeting and politeness
	M-OFF	off-topic sentences

Table 1: Sentence Types for QA Threads

To automatically label sentences in a thread with their types, we adopt a sequence labeling approach, specifically linear-chain conditional random fields (CRFs), which have shown good performance in many other tasks (Lafferty, 2001). Intuitively there is a strong dependency between adjacent sentences. For example, in our data set, 45% sentences following a greeting sentence (M-GRET) are question related sentences; 53% sentences following a question inquiry sentence (Q-INQ) are solution related sentences. The following describes our modeling approaches and features used for sentence type tagging.



### 3.1.1 Linear-chain Conditional Random Field

Linear-chain CRFs is a type of undirected graphical models. Distribution of a set of variables in undirected graphical models can be written as

$$p(x, y) = \frac{1}{Z} \prod_A \psi_A(x_A, y_A) \quad (1)$$

$Z$  is the normalization constant to guarantee valid probability distributions. CRFs is a special case of undirected graphical model in which  $\psi$  are log-linear functions:

$$\psi_A(x_A, y_A) = \exp \left\{ \sum_k \theta_{A_k} f_{A_k}(x_A, y_A) \right\} \quad (2)$$

$\theta_A$  is a real value parameter vector for feature function set  $f_A$ . In the sequence labeling task, feature functions across the sequence are often tied together. In other words, feature functions at different locations of the sequence share the same parameter vector  $\theta$ .

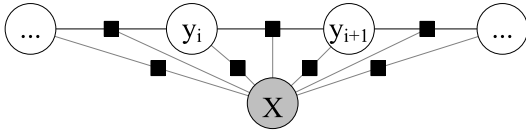


Figure 3: Graphical Structure of Linear-chain CRFs.

Linear-chain CRF is a special case of the general CRFs. In linear-chain CRF, cliques only involve two adjacent variables in the sequence. Figure 3 shows the graphical structure of a linear-chain CRF. In our case of sentence tagging, cliques only contain two adjacent sentences. Given the observation  $x$ , the probability of label sequence  $y$  is as follows:

$$p(y|x) = \frac{1}{Z} \prod_{i=1}^{|y|} \psi_e(x, y, i) \prod_{j=0}^{|y|} \psi_v(x, y, j) \quad (3)$$

$$\psi_e(x, y, i) = \exp \left\{ \sum_k \theta_{e_k} f_{e_k}(y_{i-1}, y_i, x, i) \right\} \quad (4)$$

$$\psi_v(x, y, j) = \exp \left\{ \sum_k \theta_{v_k} f_{v_k}(y_j, x, j) \right\} \quad (5)$$

where feature templates  $f_{e_k}$  and  $f_{v_k}$  correspond to edge features and node features respectively.

#### Feature Description

Cosine similarity with previous sentence.
Quote segment within two adjacent sentences?
Code segment within two adjacent sentences?
Does this sentence belong to author's post?
Is it the first sentence in a post?
Post author participated thread before?
Does the sentence contain any negative words?
Does the sentence contain any URL?
Does the sentence contain any positive words?
Does the sentence contain any question mark?
Length of the sentence.
Presence of verb.
Presence of adjective.
Sentence perplexity based on a background LM.
Bag of word features.

Table 2: Features Used in Sentence Type Tagging.

### 3.1.2 Sentence Type Tagging Features

We used various types of feature functions in sentence type tagging. Table 2 shows the complete list of features we used. Edge features are closely related to the transition between sentences. Here we use the cosine similarity between sentences, where each sentence is represented as a vector of words, with term weight calculated using TD-IDF (term frequency times inverse document frequency). High similarity between adjacent sentences suggests similar or related types. For node features, we explore different sources of information about the sentence. For example, the presence of a question mark indicates that a sentence may be a question or inquiry. Similarly, we include other cues, such as positive or negative words, verb and adjective words. Since technical forums tend to contain many system outputs, we include the perplexity of the sentence as a feature which is calculated based on a background language model (LM) learned from common English documents. We also use bag-of-word features as in many other text categorization tasks.

Furthermore, we add features to represent post level information to account for the structure of QA threads, for example, whether or not a sentence belongs to the author's post, or if a sentence is the beginning sentence of a post.

### 3.2 Sentence Dependency Tagging

Knowing only the sentence types without their dependency relations is not enough for question answering tasks. For example, correct labeling of an answer without knowing which question it actually refers to is problematic; not knowing which answer a positive or negative feedback refers to will not be helpful at all. In this section we describe how sentence dependency information is determined. Note that sentence dependency relations might not be a one-to-one relation. A many-to-many relation is also possible. Take question answer relation as an example. There could be potentially many answers spreading in many sentences, all depending on the same question. Also, it is very likely that a question is expressed in multiple sentences too.

Dependency relationship could happen between many different types of sentences, for example, answer(s) to question(s), problem clarification to question inquiry, feedback to solutions, etc. Instead of developing models for each dependency type, we treat them uniformly as dependency relations between sentences. Hence, for every two sentences, it becomes a binary classification problem, i.e., whether or not there exists a dependency relation between them. For a pair of sentences, we call the depending sentence the **source** sentence, and the depended sentence the **target** sentence. As described earlier, one source sentence can potentially depend on many different target sentences, and one target sentence can also correspond to multiple sources.

The sentence dependency task is formally defined as, given a set of sentences  $S_t$  of a thread, find the dependency relation  $\{(s, t) | s \in S_t, t \in S_t\}$ , where  $s$  is the source sentence and  $t$  is the target sentence that  $s$  depends on.

We propose two methods to find the dependency relationship. In the first approach, for each source sentence, we run a labeling procedure to find the dependent sentences. From the data, we found given a source sentence, there is strong dependency between adjacent target sentences. If one sentence is a target sentence of the source, often the next sentence is a target sentence too. In order to take advantage of such adjacent sentence dependency, we use the linear-chain CRFs for the sequence labeling. Features used in sentence dependency labeling are listed

in Table 3. Note that a lot of the node features used here are relative to the source sentence since the task here is to determine if the two sentences are related. For a thread of  $N$  sentences, we need to perform  $N$  runs of CRF labeling, one for each sentence (as the source sentence) in order to label the target sentence corresponding to this source sentence.

Feature Description	
*	Cosine similarity with previous sentence.
*	Is adjacent sentence of the same type?
*	Pair of types of the adjacent target sentences. Pair of types of the source and target sentence. Is target in the same post as the source? Do target and source belong to the same author? Cosine similarity between target and source sentence. Does target sentence happen before source? Post distance between source and target sentence.
* indicates an edge feature	

Table 3: Features Used in Sentence Dependency Labeling

The linear-chain CRFs can represent the dependency between adjacent target sentences quite well. However they cannot model the dependency between adjacent source sentences, because labeling is done for each source sentence individually. To model the dependency between both the source sentences and the target sentences, we propose to use 2D CRFs for sentence relation labeling. 2D CRFs are used in many applications considering two dimension dependencies such as object recognitions (Quattoni et al., 2004) and web information extraction (Zhu et al., 2005). The graphical structure of a 2D CRF is shown in Figure 4. Unlike one dimensional sequence labeling, a node in 2D environment is dependent on both x-axis neighbors and y-axis neighbors. In the sentence relation task, the source and target pair is a 2D relation in which its label depends on labels of both its adjacent source and its adjacent target sentence. As shown in Figure 4, looking from x-axis is the sequence of target sentences with a fixed source sentence, and from y-axis is the sequence of source sentences with a fixed target sentence. This model allows us to model all the sentence relationships jointly. 2D CRFs contain 3 templates of features: node template, x-axis edge template, and y-axis edge template. We use the same edge features and node features as in linear-chain CRFs for node features and y-axis edge features in

2D CRFs. For the x-axis edge features, we use the same feature functions as for y-axis, except that now they represent the relation between adjacent source sentences.

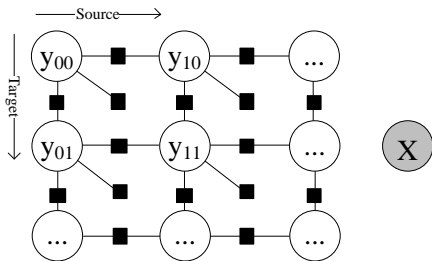


Figure 4: Graphical Structure of 2D CRF for Sentence Relation Labeling.

In a thread containing  $N$  sentences, we would have a 2D CRF containing  $N^2$  nodes in a  $N \times N$  grid. Exact inference in such a graph is intractable. In this paper we use loopy belief propagation algorithm for the inference. Loopy belief propagation is a message passing algorithm for graph inference. It calculates the marginal distribution for each node in the graph. The result is exact in some graph structures (e.g., linear-chain CRFs), and often converges to a good approximation for general graphs.

## 4 Data

We used data from ubuntu community forum general help section for the experiments and evaluation. This is a technical support section that provides answers to the latest problems in Ubuntu Linux. Among all the threads that we have crawled, we selected 200 threads for this initial study. They contain between 2 – 10 posts and at least 2 participants. Sentences inside each thread are segmented using Apache OpenNLP tools. In total, there are 706 posts and 3,483 sentences. On average, each thread contains 3.53 posts, and each post contains around 4.93 sentences. Two annotators were recruited to annotate the sentence type and the dependency relation between sentences. Annotators are both computer science department undergraduate students. They are provided with detailed explanation of the annotation standard. The distribution of sentence types in the annotated data is shown in Table 4, along with inter-annotator Kappa statistics calculated using 20

common threads annotated by both annotators. We can see that the majority of the sentences are about problem descriptions and solutions. In general, the agreement between the two annotators is quite good.

General Type	Category	Percentage	Kappa
Problems	P-STAT	12.37	0.88
	P-CONT	37.30	0.77
	P-CLRF	1.01	0.98
Answers	A-SOLU	9.94	0.89
	A-EXPL	11.60	0.89
	A-INQU	1.38	0.99
Confirmation	C-GRAT	5.06	0.98
	C-NEGA	1.98	0.96
	C-POSI	1.84	0.96
Miscellaneous	M-QCOM	1.98	0.93
	M-ACOM	1.47	0.96
	M-GRET	1.01	0.96
	M-OFF	7.92	0.96

Table 4: Distribution and Inter-annotator Agreement of Sentence Types in Data

There are in total 1,751 dependency relations identified by the annotators among those tagged sentences. Note that we are only dealing with intra-thread sentence dependency, that is, no dependency among sentences in different threads is labeled. Considering all the possible sentence pairs in each thread, the labeled dependency relations represent a small percentage. The most common dependency is problem description to problem question. This shows that users tend to provide many details of the problem. This is especially true in technical forums. Seeing questions without their context would be confusing and hard to solve. The relation of answering solutions and question dependency is also very common, as expected. The third common relation is the feedback dependency. Even though the number of feedback sentences is small in the data set, it plays a vital role to determine the quality of answers. The main reason for the small number is that, unlike problem descriptions, much fewer sentences are needed to give feedbacks.

## 5 Experiment

In the experiment, we randomly split annotated threads into three disjoint sets, and run a three-fold cross validation. Within each fold, first sentence types are labeled using linear-chain CRFs, then the

resulting sentence type tagging is used in the second pass to determine dependency relations. For part-of-speech (POS) tagging of the sentences, we used Stanford POS Tagger (Toutanova and Manning, 2000). All the graphical inference and estimations are done using MALLET package (McCallum, 2002).

In this paper, we evaluate the results using standard precision and recall. In the sentence type tagging task, we calculate precision, recall, and  $F_1$  score for each individual tag. For the dependency tagging task, a pair identified by the system is correct only if the exact pair appears in the reference annotation. Precision and recall scores are calculated accordingly.

### 5.1 Sentence Type Tagging Results

The results of sentence type tagging using linear-chain CRFs are shown in Table 5. For a comparison, we include results using a basic first-order HMM model. Because HMM is a generative model, we use only bag of word features in the generative process. The observation probability is the probability of the sentence generated by a unigram language model, trained for different sentence types. Since for some applications, fine grained categories may not be needed, for example, in the case of finding questions and answers in a thread, we also include in the table the tagging results when only the general categories are used in both training and testing.

We can see from the table that using CRFs achieves significantly better performance than HMMs for most categories, except greeting and off-topic types. This is mainly because of the advantage of CRFs, allowing the incorporation of rich discriminative features. For the two major types of problems and answers, in general, our system shows very good performance. Even for minority types like feedbacks, it also performs reasonably well. When using coarse types, the performance on average is better compared to the finer grained categories, mainly because of the fewer classes in the classification task. Using the fine grained categories, we found that the system is able to tell the difference between “problem statement” (P-STAT) and “problem context” (P-CONT). Note that in our task, a problem statement is not necessarily a question sentence. Instead it could be any sentence that expresses the need for a solu-

	Linear-chain CRF		First-order HMM	
13 Fine Grained Types				
Tag	Prec. / Rec.	$F_1$	Prec. / Rec.	$F_1$
M-GRET	0.45 / 0.58	0.51	0.73 / 0.57	<b>0.64</b>
P-STAT	0.79 / 0.72	<b>0.75</b>	0.35 / 0.34	0.35
P-CONT	0.80 / 0.74	<b>0.77</b>	0.58 / 0.18	0.27
A-INQU	0.37 / 0.48	<b>0.42</b>	0.11 / 0.25	0.15
A-SOLU	0.78 / 0.64	<b>0.71</b>	0.27 / 0.29	0.28
A-EXPL	0.4 / 0.76	<b>0.53</b>	0.24 / 0.19	0.21
M-POST	0.5 / 0.41	<b>0.45</b>	0.04 / 0.1	0.05
C-GRAT	0.43 / 0.53	<b>0.48</b>	0.01 / 0.25	0.02
M-NEGA	0.67 / 0.5	<b>0.57</b>	0.09 / 0.31	0.14
M-OFF	0.11 / 0.23	0.15	0.20 / 0.23	<b>0.21</b>
P-CLRF	0.15 / 0.33	<b>0.21</b>	0.10 / 0.12	0.11
M-ACOM	0.27 / 0.38	<b>0.32</b>	0.09 / 0.1	0.09
M-QCOM	0.34 / 0.32	<b>0.33</b>	0.08 / 0.23	0.11
4 General Types				
Tag	Prec. / Rec.	$F_1$	Prec. / Rec.	$F_1$
Problem	0.85 / 0.76	<b>0.80</b>	0.73 / 0.27	0.39
Answers	0.65 / 0.72	<b>0.68</b>	0.45 / 0.36	0.40
Confirm.	0.80 / 0.74	<b>0.77</b>	0.06 / 0.26	0.10
Misc.	0.43 / 0.61	<b>0.51</b>	0.04 / 0.36	0.08

Table 5: Sentence Type Tagging Performance Using CRFs and HMM.

tion.

We also performed some analysis of the features using the feature weights in the trained CRF models. We find that some post level information is relatively important. For example, the feature representing whether the sentence is before a “code” segment has a high weight for problem description classification. This is because in linux support forum, people usually put some machine output after their problem description. We also notice that the weights for verb words are usually high. This is intuitive since the “verb” of a sentence can often determine its purpose.

### 5.2 Sentence Dependency Tagging Results

Table 6 shows the results using linear-chain CRFs (L-CRF) and 2D CRFs for sentence dependency tagging. We use different settings in our experiments. For the categories of sentence types, we evaluate using both the fine grained (13 types) and the coarse categories (4 types). Furthermore, we examine two ways to obtain the sentence types. First, we use the output from automatic sentence type tagging. In the second one, we use the sentence type information from the human annotated data in order to avoid the error propagation from automatic sentence type la-

belonging. This gives an oracle upper bound for the second pass performance.

Using Oracle Sentence Type				
Setup		Precision	Recall	$F_1$
13 types	L-CRF	0.973	0.453	0.618
	2D-CRF	0.985	0.532	0.691
4 general	L-CRF	0.941	0.124	0.218
	2D-CRF	0.956	0.145	0.252
Using System Sentence Type				
Setup		Precision	Recall	$F_1$
13 types	L-CRF	0.943	0.362	0.523
	2D-CRF	0.973	0.394	0.561
4 general	L-CRF	0.939	0.101	0.182
	2D-CRF	0.942	0.127	0.223

Table 6: Sentence Dependency Tagging Performance

From the results we can see that 2D CRFs outperform linear-chain CRFs for all the conditions. This shows that by modeling the 2D dependency in source and target sentences, system performance is improved. For the sentence types, when using automatic sentence type tagging systems, there is a performance drop. The performance gap between using the reference and automatic sentence types suggests that there is still room for improvement from better sentence type tagging. Regarding the categories used for the sentence types, we observe that they have an impact on dependence tagging performance. When using general categories, the performance is far behind that using the fine grained types. This is because some important information is lost when grouping categories. For example, a dependency relation can be: “A-EXPL” (explanation for solutions) depends on “A-SOLU” (solutions); however, when using coarse categories, both are mapped to “Solution”, and having one “Solution” depending on another “Solution” is not very intuitive and hard to model properly. This shows that detailed category information is very important for dependency tagging even though the tagging accuracy from the first pass is far from perfect.

Currently our system does not put constraints on the sentence types for which dependencies exist. In the system output we find that sometimes there are obvious dependency errors, such as a positive feedback depending on a negative feedback. We may improve our models by taking into account different sentence types and dependency relations.

## 6 Conclusion

In this paper, we investigated sentence dependency tagging of question and answer (QA) threads in on-line forums. We define the thread tagging task as a two-step process. In the first step, sentence types are labeled. We defined 13 sentence types in order to capture rich information of sentences to benefit question answering systems. Linear chain CRF is used for sentence type tagging. In the second step, we label actual dependency between sentences. First, we propose to use a linear-chain CRF to label possible target sentences for each source sentence. Then we improve the model to consider the dependency between sentences along two dimensions using a 2D CRF. Our experiments show promising performance in both tasks. This provides a good pre-processing step towards automatic question answering. In the future, we plan to explore using constrained CRF for more accurate dependency tagging. We will also use the result from this work in other tasks such as answer quality ranking and answer summarization.

## 7 Acknowledgment

This work is supported by DARPA under Contract No. HR0011-12-C-0016 and NSF No. 0845484. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of DARPA or NSF.

## References

- Kristy Elizabeth Boyer, Robert Phillips, Eun Young Ha, Michael D. Wallis, Mladen A. Vouk, and James C. Lester. 2009. Modeling dialogue structure with adjacency pair analysis and hidden markov models. In *Proc. NAACL-Short*, pages 49–52.
- Alexander Clark and Andrei Popescu-Belis. 2004. Multi-level dialogue act tags. In *Proc. SIGDIAL*, pages 163–170.
- Gao Cong, Long Wang, Chinyew Lin, Youngin Song, and Yueheng Sun. 2008. Finding question-answer pairs from online forums. In *Proc. SIGIR*, pages 467–474.
- Shilin Ding, Gao Cong, Chinyew Lin, and Xiaoyan Zhu. 2008. Using conditional random fields to extract contexts and answers of questions from online forums. In *Proc. ACL-HLT*.
- Gang Ji and J Bilmes. 2005. Dialog Act Tagging Using Graphical Models. In *Proc. ICASSP*.

- John Lafferty. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, pages 282–289.
- Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu>.
- Ariadna Quattoni, Michael Collins, and Trevor Darrell. 2004. Conditional random fields for object recognition. In *Proc. NIPS*, pages 1097–1104.
- Lokesh Shrestha and Kathleen Mckeown. 2004. Detection of question-answer pairs in email conversations. In *Proc. Coling*, pages 889–895.
- Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26:339–373.
- Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proc. EMNLP/VLC*, pages 63–70.
- Jun Zhu, Zaiqing Nie, Ji R. Wen, Bo Zhang, and Wei Y. Ma. 2005. 2D Conditional Random Fields for Web information extraction. In *Proc. ICML*, pages 1044–1051, New York, NY, USA.

# Mining Entity Types from Query Logs via User Intent Modeling

**Patrick Pantel**  
Microsoft Research  
One Microsoft Way  
Redmond, WA 98052, USA  
ppantel@microsoft.com

**Thomas Lin**  
Computer Science & Engineering  
University of Washington  
Seattle, WA 98195, USA  
tlin@cs.washington.edu

**Michael Gamon**  
Microsoft Research  
One Microsoft Way  
Redmond, WA 98052, USA  
mgamon@microsoft.com

## Abstract

We predict entity type distributions in Web search queries via probabilistic inference in graphical models that capture how entity-bearing queries are generated. We jointly model the interplay between latent user intents that govern queries and unobserved entity types, leveraging observed signals from query formulations and document clicks. We apply the models to resolve entity types in new queries and to assign prior type distributions over an existing knowledge base. Our models are efficiently trained using maximum likelihood estimation over millions of real-world Web search queries. We show that modeling user intent significantly improves entity type resolution for head queries over the state of the art, on several metrics, without degradation in tail query performance.

## 1 Introduction

Commercial search engines are providing ever-richer experiences around entities. Querying for a dish on Google yields recipe filters such as cook time, calories, and ingredients. Querying for a movie on Yahoo triggers user ratings, cast, tweets and showtimes. Bing further allows the movie to be directly added to the user's Netflix queue. Entity repositories such as Freebase, IMDB, Facebook Pages, Factual, Pricegrabber, and Wikipedia are increasingly leveraged to enable such experiences.

There are, however, inherent problems in the entity repositories: (a) coverage: although coverage of head entity types is often reliable, the tail can be sparse; (b) noise: created by spammers, extraction

errors or errors in crowdsourced content; (c) ambiguity: multiple types or entity identifiers are often associated with the same surface string; and (d) over-expression: many entities have types that are never used in the context of Web search.

There is an opportunity to automatically tailor knowledge repositories to the Web search scenario. Desirable capabilities of such a system include: (a) determining the prior type distribution in Web search for each entity in the repository; (b) assigning a type distribution to new entities; (c) inferring the correct sense of an entity in a particular query context; and (d) adapting to a search engine and time period.

In this paper, we build such a system by leveraging Web search usage logs with large numbers of user sessions seeking or transacting on entities. We cast the task as performing probabilistic inference in a graphical model that captures how queries are generated, and then apply the model to contextually recognize entity types in new queries. We motivate and design several generative models based on the theory that search users' (unobserved) intents govern the types of entities, the query formulations, and the ultimate clicks on Web documents. We show that jointly modeling user intent and entity type significantly outperforms the current state of the art on the task of entity type resolution in queries. The major contributions of our research are:

- We introduce the idea that latent user intents can be an important factor in modeling type distributions over entities in Web search.
- We propose generative models and inference procedures using signals from query context, click, entity, entity type, and user intent.

- We propose an efficient learning technique and a robust implementation of our models, using real-world query data, and a realistic large set of entity types.
- We empirically show that our models outperform the state of the art and that modeling latent intent contributes significantly to these results.

## 2 Related Work

### 2.1 Finding Semantic Classes

A closely related problem is that of finding the semantic classes of entities. Automatic techniques for finding semantic classes include unsupervised clustering (Schütze, 1998; Pantel and Lin, 2002), hyponym patterns (Hearst, 1992; Pantel et al., 2004; Kozareva et al., 2008), extraction patterns (Etzioni et al., 2005), hidden Markov models (Ritter et al., 2009), classification (Rahman and Ng, 2010) and many others. These techniques typically leverage large corpora, while projects such as WordNet (Miller et al., 1990) and Freebase (Bollacker et al., 2008) have employed editors to manually enumerate words and entities with their semantic classes.

The aforementioned methods do not use query logs or explicitly determine the relative probabilities of different entity senses. A method might learn that there is independently a high chance of *eBay* being a *website* and an *employer*, but does not specify which usage is more common. This is especially problematic, for example, if one wishes to leverage Freebase but only needs the most commonly used senses (e.g., *Al Gore* is a US Vice President), rather than all possible obscure senses (Freebase contains 30+ senses, including ones such as *Impersonated Celebrity* and *Quotation Subject*). In scenarios such as this, our proposed method can increase the usability of systems that find semantic classes. We also expand upon text corpora methods in that the type priors can adapt to Web search signals.

### 2.2 Query Log Mining

Query logs have traditionally been mined to improve search (Baeza-Yates et al., 2004; Zhang and Nasraoui, 2006), but they can also be used in place of (or in addition to) text corpora for learning semantic classes. Query logs can contain billions of en-

tries, they provide an independent signal from text corpora, their timestamps allow the learning of type priors at specific points in time, and they can contain information such as clickthroughs that are not found in text corpora. Sekine and Suzuki (2007) used frequency features on context words in query logs to learn semantic classes of entities. Paşca (2007) used extraction techniques to mine instances of semantic classes from query logs. Rüd et al. (2011) found that cross-domain generalizations learned from Web search results are applicable to NLP tasks such as NER. Alfonseca et al. (2010) mined query logs to find attributes of entity instances. However, these projects did not learn relative probabilities of different senses.

### 2.3 User Intents in Search

Learning from query logs also allows us to leverage the concept of *user intents*. When users submit search queries, they often have specific intents in mind. Broder (2002) introduced 3 top level intents: *Informational* (e.g., wanting to learn), *Navigational* (wanting to visit a site), and *Transactional* (e.g., wanting to buy/sell). Rose and Levinson (2004) further divided these into finer-grained subcategories, and Yin and Shah (2010) built hierarchical taxonomies of search intents. Jansen et al. (2007), Hu et al. (2009), and Radlinski et al. (2010) examined how to infer the intent of queries. We are not aware of any other work that has leveraged user intents to learn type distributions.

### 2.4 Topic Modeling on Query Logs

The closest work to ours is Guo et al.’s (2009) research on Named Entity Recognition in Queries. Given an entity-bearing query, they attempt to identify the entity and determine the type posteriors. Our work significantly scales up the type posteriors component of their work. While they only have four potential types (*Movie*, *Game*, *Book*, *Music*) for each entity, we employ over 70 popular types, allowing much greater coverage of real entities and their types. Because they only had four types, they were able to hand label their training data. In contrast, our system self-labels training examples by searching query logs for high-likelihood entities, and must handle any errors introduced by this process. Our models also expand upon theirs by jointly modeling



entity type with latent user intents, and by incorporating click signals.

Other projects have also demonstrated the utility of topic modeling on query logs. Carman et al. (2010) modeled users and clicked documents to personalize search results and Gao et al. (2011) applied topic models to query logs in order to improve document ranking for search.

### 3 Joint Model of Types and User Intents

We turn our attention now to the task of mining the type distributions of entities and of resolving the type of an entity in a particular query context. Our approach is to probabilistically describe how entity-bearing queries are generated in Web search. We theorize that search queries are governed by a latent user intent, which in turn influences the entity types, the choice of query words, and the clicked hosts. We develop inference procedures to infer the prior type distributions of entities in Web search as well as to resolve the type of an entity in a query, by maximizing the probability of observing a large collection of real-world queries and their clicked hosts.

We represent a query  $q$  by a triple  $\{n_1, e, n_2\}$ , where  $e$  represents the entity mentioned in the query,  $n_1$  and  $n_2$  are respectively the pre- and post-entity contexts (possibly empty), referred to as *refiners*. Details on how we obtain our corpus are presented in Section 4.2.

#### 3.1 Intent-based Model (IM)

In this section we describe our main model, **IM**, illustrated in Figure 1. We derive a learning algorithm for the model in Section 3.2 and an inference procedure in Section 3.3.

Recall our discussion of intents from Section 2.3. The unobserved semantic type of an entity  $e$  in a query is strongly correlated with the unobserved user intent. For example, if a user queries for “*song*”, then she is likely looking to ‘listen to it’, ‘download it’, ‘buy it’, or ‘find lyrics’ for it. Our model incorporates this user intent as a latent variable.

The choice of the query refiner words,  $n_1$  and  $n_2$ , is also clearly influenced by the user intent. For example, refiners such as “lyrics” and “words” are more likely to be used in queries where the intent is

<p>For each query/click pair <math>\{q, c\}</math>  type <math>t \sim \text{Multinomial}(\tau)</math>  intent <math>i \sim \text{Multinomial}(\theta_t)</math>  entity <math>e \sim \text{Multinomial}(\psi_i)</math>  switch <math>s_1 \sim \text{Bernoulli}(\sigma_i)</math>  switch <math>s_2 \sim \text{Bernoulli}(\sigma_i)</math>  if <math>(s_1)</math> l-context <math>n_1 \sim \text{Multinomial}(\phi_i)</math>  if <math>(s_2)</math> r-context <math>n_2 \sim \text{Multinomial}(\phi_i)</math>  click <math>c \sim \text{Multinomial}(\omega_i)</math></p>
---

Table 1: Model **IM**: Generative process for entity-bearing queries.

to ‘find lyrics’ than in queries where the intent is to ‘listen’. The same is true for clicked hosts: clicks on “lyrics.com” and “songlyrics.com” are more likely to occur when the intent is to ‘find lyrics’, whereas clicks on “pandora.com” and “last.fm” are more likely for a ‘listen’ intent.

Model **IM** leverages each of these signals: latent intent, query refiners, and clicked hosts. It generates entity-bearing queries by first generating an entity type, from which the user intent and entity is generated. In turn, the user intent is then used to generate the query refiners and the clicked host. In our data analysis, we observed that over 90% of entity-bearing queries did not contain any refiner words  $n_1$  and  $n_2$ . In order to distribute more probability mass to non-empty context words, we explicitly represent the empty context using a switch variable that determines whether a context will be empty.

The generative process for **IM** is described in Table 1. Consider the query “*ymca lyrics*”. Our model first generates the type `song`, then given the type it generates the entity “*ymca*” and the intent ‘find lyrics’. The intent is then used to generate the pre- and post-context words  $\emptyset$  and “*lyrics*”, respectively, and a click on a host such as “*lyrics.com*”.

For mathematical convenience, we assume that the user intent is generated independently of the entity itself. Without this assumption, we would require learning a parameter for each intent-type-entity configuration, exploding the number of parameters. Instead, we choose to include these dependencies at the time of inference, as described later.

Recall that  $q = \{n_1, e, n_2\}$  and let  $s = \{s_1, s_2\}$ , where  $s_1 = 1$  if  $n_1$  is not empty and  $s_2 = 1$  if  $n_2$  is not empty, 0 otherwise. The joint probability of the model is the product of the conditional distributions, as given by:

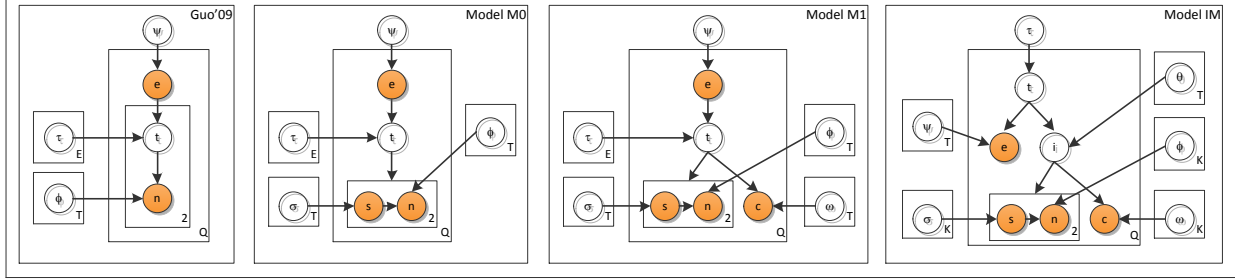


Figure 1: Graphical models for generating entity-bearing queries. **Guo’09** represents the current state of the art (Guo et al., 2009). Models **M0** and **M1** add an empty context switch and click information, respectively. Model **IM** further constrains the query by the latent user intent.

$$P(t, i, q, c | \tau, \Theta, \Psi, \sigma, \Phi, \Omega) = P(t | \tau)P(i | t, \Theta)P(e | t, \Psi)P(c | i, \Omega) \prod_{j=1}^2 P(n_j | i, \Phi)^{I[s_j=1]}P(s_j | i, \sigma)$$

We now define each of the terms in the joint distribution. Let  $T$  be the number of entity types. The probability of generating a type  $t$  is governed by a multinomial with probability vector  $\tau$ :

$$P(t=\hat{t}) = \prod_{j=1}^T \tau_j^{I[j=\hat{t}]}, \text{ s.t. } \sum_{j=1}^T \tau_j = 1$$

where  $I$  is an indicator function set to 1 if its condition holds, and 0 otherwise.

Let  $K$  be the number of latent user intents that govern our query log, where  $K$  is fixed in advance. Then, the probability of intents  $i$  is defined as a multinomial distribution with probability vector  $\theta_t$  such that  $\Theta = [\theta_1, \theta_2, \dots, \theta_T]$  captures the matrix of parameters across all  $T$  types:

$$P(i=\hat{i} | t=\hat{t}) = \prod_{j=1}^K \Theta_{\hat{i},j}^{I[j=\hat{i}]}, \text{ s.t. } \forall t \sum_{j=1}^K \Theta_{t,j} = 1$$

Let  $E$  be the number of known entities. The probability of generating an entity  $e$  is similarly governed by a parameter  $\Psi$  across all  $T$  types:

$$P(e=\hat{e} | t=\hat{t}) = \prod_{j=1}^E \Psi_{\hat{t},j}^{I[j=\hat{e}]}, \text{ s.t. } \forall t \sum_{j=1}^E \Psi_{t,j} = 1$$

The probability of generating an empty or non-empty context  $s$  given intent  $i$  is given by a Bernoulli with parameter  $\sigma_i$ :

$$P(s | i=\hat{i}) = \sigma_{\hat{i}}^{I[s=1]}(1 - \sigma_{\hat{i}})^{I[s=0]}$$

Let  $V$  be the shared vocabulary size of all query refiner words  $n_1$  and  $n_2$ . Given an intent,  $i$ , the probability of generating a refiner  $n$  is given by a multinomial distribution with probability vector  $\phi_i$  such that  $\Phi = [\phi_1, \phi_2, \dots, \phi_K]$  represents parameters across intents:

$$P(n=\hat{n} | i=\hat{i}) = \prod_{v=1}^V \Phi_{\hat{i},v}^{I[v=\hat{n}]}, \text{ s.t. } \forall i \sum_{v=1}^V \Phi_{i,v} = 1$$

Finally, we assume there are  $H$  possible click values, corresponding to  $H$  Web hosts. A click on a host is similarly determined by an intent  $i$  and is governed by parameter  $\Omega$  across all  $K$  intents:

$$P(c=\hat{c} | i=\hat{i}) = \prod_{h=1}^H \Omega_{\hat{i},h}^{I[h=\hat{c}]}, \text{ s.t. } \forall i \sum_{h=1}^H \Omega_{i,h} = 1$$

## 3.2 Learning

Given a query corpus  $\mathcal{Q}$  consisting of  $N$  independently and identically distributed queries  $q^j = \{n_1^j, e^j, n_2^j\}$  and their corresponding clicked hosts  $c^j$ , we estimate the parameters  $\tau, \Theta, \Psi, \sigma, \Phi$ , and  $\Omega$  by maximizing the (log) probability of observing  $\mathcal{Q}$ . The log  $P(\mathcal{Q})$  can be written as:

$$\log P(\mathcal{Q}) = \sum_{j=1}^N \sum_{t,i} P^j(t, i | q, c) \log P^j(q, c, t, i)$$

In the above equation,  $P^j(t, i | q, c)$  is the posterior distribution over types and user intents for the  $j^{\text{th}}$  query. We use the Expectation-Maximization (EM) algorithm to estimate the parameters. The parameter updates are obtained by computing the derivative of  $\log P(\mathcal{Q})$  with respect to each parameter, and setting the resultant to 0.

The update for  $\tau$  is given by the average of the posterior distributions over the types:

$$\tau_{\hat{t}} = \frac{\sum_{j=1}^N \sum_i P^j(t=\hat{t}, i | q, c)}{\sum_{j=1}^N \sum_{t,i} P^j(t, i | q, c)}$$

For a fixed type  $t$ , the update for  $\theta_t$  is given by the weighted average of the latent intents, where the weights are the posterior distributions over the types, for each query:

$$\Theta_{\hat{t}, \hat{i}} = \frac{\sum_{j=1}^N P^j(t=\hat{t}, i=\hat{i} | q, c)}{\sum_{j=1}^N \sum_i P^j(t=\hat{t}, i | q, c)}$$

Similarly, we can update  $\Psi$ , the parameters that govern the distribution over entities for each type:

$$\Psi_{\hat{t}, \hat{e}} = \frac{\sum_{j=1}^N \sum_i P^j(t=\hat{t}, i | q, c) I[e^j=\hat{e}]}{\sum_{j=1}^N \sum_i P^j(t=\hat{t}, i | q, c)}$$

Now, for a fixed user intent  $i$ , the update for  $\omega_i$  is given by the weighted average of the clicked hosts, where the weights are the posterior distributions over the intents, for each query:

$$\Omega_{\hat{i}, \hat{c}} = \frac{\sum_{j=1}^N \sum_t P^j(t, i=\hat{i} | q, c) I[c^j=\hat{c}]}{\sum_{j=1}^N \sum_t P^j(t, i=\hat{i} | q, c)}$$

Similarly, we can update  $\Phi$  and  $\sigma$ , the parameters that govern the distribution over query refiners and empty contexts for each intent, as:

$$\Phi_{\hat{i}, \hat{n}} = \frac{\sum_{j=1}^N \sum_t P^j(t, i=\hat{i} | q, c) \left[ I[n_1^j=\hat{n}] I[s_1^j=1] + I[n_2^j=\hat{n}] I[s_2^j=1] \right]}{\sum_{j=1}^N \sum_t P^j(t, i=\hat{i} | q, c) \left[ I[s_1^j=1] + I[s_2^j=1] \right]}$$

and

$$\sigma_{\hat{i}} = \frac{\sum_{j=1}^N \sum_t P^j(t, i=\hat{i} | q, c) \left[ I[s_1=1] + I[s_2=1] \right]}{2 \sum_{j=1}^N \sum_t P^j(t, i=\hat{i} | q, c)}$$

### 3.3 Decoding

Given a query/click pair  $\{q, c\}$ , and the learned IM model, we can apply Bayes' rule to find the posterior distribution,  $P(t, i | q, c)$ , over the types and intents, as it is proportional to  $P(t, i, q, c)$ . We compute this quantity exactly by evaluating the joint for each combination of  $t$  and  $i$ , and the observed values of  $q$  and  $c$ .

It is important to note that at runtime when a new query is issued, we have to resolve the entity in the absence of any observed click. However, we do have access to historical click probabilities,  $P(c | q)$ .

We use this information to compute  $P(t | q)$  by marginalizing over  $i$  as follows:

$$P(t | q) = \sum_i \sum_{j=1}^H P(t, i | q, c_j) P(c_j | q) \quad (1)$$

### 3.4 Comparative Models

Figure 1 also illustrates the current state-of-the-art model **Guo'09** (Guo et al., 2009), described in Section 2.4, which utilizes only query refinement words to infer entity type distributions. Two extensions to this model that we further study in this paper are also shown: Model **M0** adds the empty context switch parameter and Model **M1** further adds click information. In the interest of space, we omit the update equations for these models, however they are trivial to adapt from the derivations of Model **IM** presented in Sections 3.1 and 3.2.

### 3.5 Discussion

**Full Bayesian Treatment:** In the above models, we learn point estimates for the parameters  $(\tau, \Theta, \Psi, \sigma, \Phi, \Omega)$ . One can take a Bayesian approach and treat these parameters as variables (for instance, with Dirichlet and Beta prior distributions), and perform Bayesian inference. However, exact inference will become intractable and we would need to resort to methods such as variational inference or sampling. We found this extension unnecessary, as we had a sufficient amount of training data to estimate all parameters reliably. In addition, our approach enabled us to learn (and perform inference in) the model with large amounts of data with reasonable computing time.

**Fitting to an existing Knowledge Base:** Although in general our model decodes type distributions for arbitrary entities, in many practical cases it is beneficial to constrain the types to those admissible in a fixed knowledge base (such as Freebase). As an example, if the entity is "ymca", admissible types may include `song`, `place`, and `educational_institution`. When resolving types, during inference, one can restrict the search space to only these admissible types. A desirable side effect of this strategy is that only valid ambiguities are captured in the posterior distribution.

## 4 Evaluation Methodology

We refer to *QL* as a set of English Web search queries issued to a commercial search engine over a period of several months.

### 4.1 Entity Inventory

Although our models generalize to any entity repository, we experiment in this paper with entities covering a wide range of web search queries, coming from 73 types in Freebase. We arrived at these types by grepping for all entities in Freebase within *QL*, following the procedure described in Section 4.2, and then choosing the top most frequent types such that 50% of the queries are covered by an entity of one of these types<sup>1</sup>.

### 4.2 Training Data Construction

In order to learn type distributions by jointly modeling user intents and a large number of types, we require a large set of training examples containing tagged entities and their potential types. Unlike in Guo et al. (2009), we need a method to automatically label *QL* to produce these training cases since manual annotation is impossible for the range of entities and types that we consider. Reliably recognizing entities in queries is not a solved problem. However, for training we do not require high coverage of entities in *QL*, so high precision on a sizeable set of query instances can be a proper proxy.

To this end, we collect candidate entities in *QL* via simple string matching on Freebase entity strings within our preselected 73 types. To achieve high precision from this initial (high-recall, low-precision) candidate set we use a number of heuristics to only retain highly likely entities. The heuristics include retaining only matches on entities that appear capitalized more than 50% in their occurrences in Wikipedia. Also, a standalone score filter (Jain and Pennacchiotti, 2011) of 0.9 is used, which is based on the ratio of string occurrence as

---

<sup>1</sup>In this process, we omitted any non-core Freebase type (e.g., `/user/*` and `/base/*`), types used for representation (e.g., `/common/*` and `/type/*`), and too general types (e.g., `/people/person` and `/location/location`) identified by if a type contains multiple other prominent subtypes. Finally, we conflated seven of the types that overlapped with each other into four types (such as `/book/written_work` and `/book/book`).

an exact match in queries to how often it occurs as a partial match.

The resulting queries are further filtered by keeping only those where the pre- and post-entity contexts ( $n_1$  and  $n_2$ ) were empty or a single word (accounting for a very large fraction of the queries). We also eliminate entries with clicked hosts that have been clicked fewer than 100 times over the entire *QL*. Finally, for training we filter out any query with an entity that has more than two potential types<sup>2</sup>. This step is performed to reduce recognition errors by limiting the number of potential ambiguous matches. We experimented with various thresholds on allowable types and settled on the value two.

The resulting training data consists of several million queries, 73 different entity types, and approximately 135K different entities, 100K different refiner words, and 40K clicked hosts.

### 4.3 Test Set Annotation

We sampled two datasets, *HEAD* and *TAIL*, each consisting of 500 queries containing an entity belonging to one of the 73 types in our inventory, from a frequency-weighted random sample and a uniform random sample of *QL*, respectively.

We conducted a user study to establish a gold standard of the correct entity types in each query. A total of seven different independent and paid professional annotators participated in the study. For each query in our test sets, we displayed the query, associated clicked host, and entity to the annotator, along with a list of permissible types from our type inventory. The annotator is tasked with identifying all applicable types from that list, or marking the test case as faulty because of an error in entity identification, bad click host (e.g. dead link) or bad query (e.g. non-English). This resulted in 2,092 test cases (`{query, entity, type}`-tuples). Each test case was annotated by two annotators. Inter-annotator agreement as measured by Fleiss'  $\kappa$  was 0.445 (0.498 on *HEAD* and 0.386 on *TAIL*), considered moderate agreement.

From *HEAD* and *TAIL*, we eliminated three categories of queries that did not offer any interesting type disambiguation opportunities:

- queries that contained entities with only one

---

<sup>2</sup>For testing we did not omit any entity or type.

	HEAD				TAIL			
	nDCG	MAP	MAP <sub>w</sub>	Prec@1	nDCG	MAP	MAP <sub>w</sub>	Prec@1
<b>B<sub>FB</sub></b>	0.71	0.60	0.45	0.30	0.73	0.64	0.49	0.35
<b>Guo'09</b>	0.79 <sup>†</sup>	0.71 <sup>†</sup>	0.62 <sup>†</sup>	0.51 <sup>†</sup>	<b>0.80<sup>†</sup></b>	<b>0.73<sup>†</sup></b>	<b>0.66<sup>†</sup></b>	<b>0.52<sup>†</sup></b>
<b>M0</b>	0.79 <sup>†</sup>	0.72 <sup>†</sup>	0.65 <sup>†</sup>	0.52 <sup>†</sup>	<b>0.82<sup>†</sup></b>	<b>0.75<sup>†</sup></b>	<b>0.67<sup>†</sup></b>	<b>0.57<sup>†</sup></b>
<b>M1</b>	0.83 <sup>‡</sup>	0.76 <sup>‡</sup>	0.72 <sup>‡</sup>	0.61 <sup>‡</sup>	<b>0.81<sup>†</sup></b>	<b>0.74<sup>†</sup></b>	<b>0.67<sup>†</sup></b>	<b>0.55<sup>†</sup></b>
<b>IM</b>	<b>0.87<sup>‡</sup></b>	<b>0.82<sup>‡</sup></b>	<b>0.77<sup>‡</sup></b>	<b>0.73<sup>‡</sup></b>	<b>0.80<sup>†</sup></b>	<b>0.72<sup>†</sup></b>	<b>0.66<sup>†</sup></b>	<b>0.52<sup>†</sup></b>

Table 2: Model analysis on HEAD and TAIL. <sup>†</sup> indicates statistical significance over **B<sub>FB</sub>**, and <sup>‡</sup> over both **B<sub>FB</sub>** and **Guo'09**. Bold indicates statistical significance over all non-bold models in the column. Significance is measured using the Student's *t*-test at 95% confidence.

potential type from our inventory;

- queries where the annotators rated all potential types as good; and
- queries where judges rated none of the potential types as good

The final test sets consist of 105 head queries with 359 judged entity types and 98 tail queries with 343 judged entity types.

#### 4.4 Metrics

Our task is a ranking task and therefore the classic IR metrics **nDCG** (normalized discounted cumulative gain) and **MAP** (mean average precision) are applicable (Manning et al., 2008).

Both **nDCG** and **MAP** are sensitive to the rank position, but not the score (probability of a type) associated with each rank,  $S(r)$ . We therefore also evaluate a weighted mean average precision score **MAP<sub>w</sub>**, which replaces the precision component of **MAP**,  $P(r)$ , for the  $r^{th}$  ranked type by:

$$P(r) = \frac{\sum_{\hat{r}=1}^r I(\hat{r})S(\hat{r})}{\sum_{\hat{r}=1}^r S(\hat{r})} \quad (2)$$

where  $I(r)$  indicates if the type at rank  $r$  is judged correct.

Our fourth metric is **Prec@1**, i.e. the precision of only the top-ranked type of each query. This is especially suitable for applications where a single sense must be determined.

#### 4.5 Model Settings

We trained all models in Figure 1 using the training data from Section 4.2 over 100 EM iterations, with two folds per model. For Model **IM**, we varied the number of user intents ( $K$ ) in intervals from 100 to 400 (see Figure 3), under the assumption that multiple intents would exist per entity type.

We compare our results against two baselines.

The first baseline is an assignment of Freebase types according to their frequency in our query set **B<sub>FB</sub>**, and the second is Model **Guo'09** (Guo et al., 2009) illustrated in Figure 1.

## 5 Experimental Results

Table 2 lists the performance of each model on the *HEAD* and *TAIL* sets over each metric defined in Section 4.4. On head queries, the addition of the empty context parameter  $\sigma$  and click signal  $\Omega$  together (Model **M1**) significantly outperforms both the baseline and the state-of-the-art model **Guo'09**. Further modeling the user intent in Model **IM** results in significantly better performance over all models and across all metrics. Model **IM** shows its biggest gains in the first position of its ranking as evidenced by the **Prec@1** metric.

We observe a different behavior on tail queries where all models significantly outperform the baseline **B<sub>FB</sub>**, but are not significantly different from each other. In short, the strength of our proposed model is in improving performance on the head at no noticeable cost in the tail.

We separately tested the effect of adding the empty context parameter  $\sigma$ . Figure 2 illustrates the result on the *HEAD* data. Across all metrics,  $\sigma$  improved performance over all models<sup>3</sup>. The more expressive models benefitted more than the less expressive ones.

Table 2 reports results for Model **IM** using  $K = 200$  user intents. This was determined by varying  $K$  and selecting the top-performing value. Figure 3 illustrates the performance of Model **IM** with different values of  $K$  on the *HEAD*.

<sup>3</sup>Note that model **M0** is just the addition of the  $\sigma$  parameter over **Guo'09**.

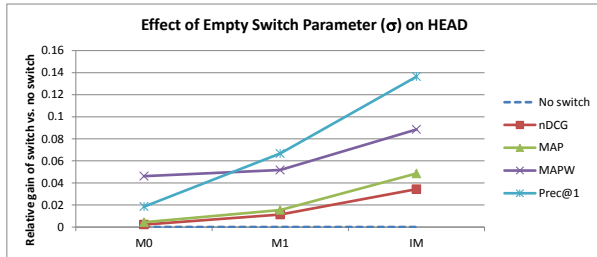


Figure 2: The switch parameter  $\sigma$  improves performance of every model and metric.

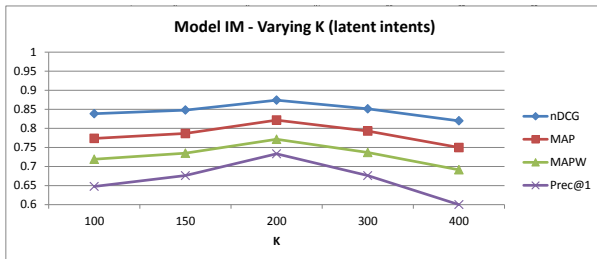


Figure 3: Model performance vs. the number of latent intents (K).

Our models can also assign a prior type distribution to each entity by further marginalizing Eq. 1 over query contexts  $n_1$  and  $n_2$ . We measured the quality of our learned type priors using the subset of queries in our *HEAD* test set that consisted of only an entity without any refiners. The results for Model IM were:  $nDCG = 0.86$ ,  $MAP = 0.80$ ,  $MAP_W = 0.75$ , and  $Prec@1 = 0.70$ . All metrics are statistically significantly better than  $B_{FB}$ ,  $Guo'09$  and M0, with 95% confidence. Compared to Model M1, Model IM is statistically significantly better on  $Prec@1$  and not significantly different on the other metrics.

**Discussion and Error Analysis:** Contrary to our results, we had expected improvements for both *HEAD* and *TAIL*. Inspection of the *TAIL* queries revealed that entities were greatly skewed towards people (e.g., actor, author, and politician). Analysis of the latent user intent parameter  $\Theta$  in Model IM showed that most people types had most of their probability mass assigned to the same three generic and common intents for people types: ‘see pictures of’, ‘find biographical information about’, and ‘see video of’. In other words, latent intents in Model IM are over-expressive and they do not help in differentiating people types.

The largest class of errors came from queries bearing an entity with semantically very similar types where our highest ranked type was not judged correct by the annotators. For example, for the query “*philippine daily inquirer*” our system ranked newspaper ahead of periodical but a judge rejected the former and approved the latter. For “*ikea catalogue*”, our system ranked magazine ahead of periodical, but again a judge rejected magazine in favor of periodical.

An interesting success case in the *TAIL* is highlighted by two queries involving the entity “*ymca*”, which in our data can either be a song, place, or educational institution. Our system learns the following priors: 0.63, 0.29, and 0.08, respectively. For the query “*jamestown ymca ny*”, IM correctly classified “*ymca*” as a place and for the query “*ymca palomar*” it correctly classified it as an educational institution. We further issued the query “*ymca lyrics*” and the type song was then highest ranked.

Our method is generalizable to any entity collection. Since our evaluation focused on the Freebase collection, it remains an open question how noise level, coverage, and breadth in a collection will affect our model performance. Finally, although we do not formally evaluate it, it is clear that training our model on different time spans of queries should lead to type distributions adapted to that time period.

## 6 Conclusion

Jointly modeling the interplay between the underlying user intents and entity types in web search queries shows significant improvements over the current state of the art on the task of resolving entity types in head queries. At the same time, no degradation in tail queries is observed. Our proposed models can be efficiently trained using an EM algorithm and can be further used to assign prior type distributions to entities in an existing knowledge base and to insert new entities into it.

Although this paper leverages latent intents in search queries, it stops short of understanding the nature of the intents. It remains an open problem to characterize and enumerate intents and to identify the types of queries that benefit most from intent models.

## References

- Enrique Alfonseca, Marius Pasca, and Enrique Robledo-Arnuncio. 2010. Acquisition of instance attributes via labeled and related instances. In *Proceedings of SIGIR-10*, pages 58–65, New York, NY, USA.
- Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. 2004. Query recommendation using query logs in search engines. In *EDBT Workshops, Lecture Notes in Computer Science*, pages 588–596. Springer.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of SIGMOD '08*, pages 1247–1250, New York, NY, USA.
- Andrei Broder. 2002. A taxonomy of web search. *SIGIR Forum*, 36:3–10.
- Mark James Carman, Fabio Crestani, Morgan Harvey, and Mark Baillie. 2010. Towards query log based personalization using topic models. In *CIKM '10*, pages 1849–1852.
- Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. volume 165, pages 91–134.
- Jianfeng Gao, Kristina Toutanova, and Wen-tau Yih. 2011. Clickthrough-based latent semantic models for web search. In *Proceedings of SIGIR '11*, pages 675–684, New York, NY, USA. ACM.
- Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. 2009. Named entity recognition in query. In *Proceedings of SIGIR-09*, pages 267–274, New York, NY, USA. ACM.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, pages 539–545.
- Jian Hu, Gang Wang, Frederick H. Lochovsky, Jian tao Sun, and Zheng Chen. 2009. Understanding user's query intent with wikipedia. In *WWW*, pages 471–480.
- Alpa Jain and Marco Pennacchiotti. 2011. Domain-independent entity extraction from web search query logs. In *Proceedings of WWW '11*, pages 63–64, New York, NY, USA. ACM.
- Bernard J. Jansen, Danielle L. Booth, and Amanda Spink. 2007. Determining the user intent of web search engine queries. In *Proceedings of WWW '07*, pages 1149–1150, New York, NY, USA. ACM.
- Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proceedings of ACL*.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.
- George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Wordnet: An on-line lexical database. volume 3, pages 235–244.
- Marius Paşca. 2007. Weakly-supervised discovery of named entities using web search queries. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, CIKM '07*, pages 683–690, New York, NY, USA. ACM.
- Patrick Pantel and Dekang Lin. 2002. Discovering word senses from text. In *SIGKDD*, pages 613–619, Edmonton, Canada.
- Patrick Pantel, Deepak Ravichandran, and Eduard Hovy. 2004. Towards terascale knowledge acquisition. In *COLING*, pages 771–777.
- Filip Radlinski, Martin Szummer, and Nick Craswell. 2010. Inferring query intent from reformulations and clicks. In *Proceedings of the 19th international conference on World wide web, WWW '10*, pages 1171–1172, New York, NY, USA. ACM.
- Altaf Rahman and Vincent Ng. 2010. Inducing fine-grained semantic classes via hierarchical and collective classification. In *Proceedings of COLING*, pages 931–939.
- Alan Ritter, Stephen Soderland, and Oren Etzioni. 2009. What is this, anyway: Automatic hypernym discovery. In *Proceedings of AAAI-09 Spring Symposium on Learning by Reading and Learning to Read*, pages 88–93.
- Daniel E. Rose and Danny Levinson. 2004. Understanding user goals in web search. In *Proceedings of the 13th international conference on World Wide Web, WWW '04*, pages 13–19, New York, NY, USA. ACM.
- Stefan Rüd, Massimiliano Ciaramita, Jens Müller, and Hinrich Schütze. 2011. Piggyback: Using search engines for robust cross-domain named entity recognition. In *Proceedings of ACL '11*, pages 965–975, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24:97–123, March.
- Satoshi Sekine and Hisami Suzuki. 2007. Acquiring ontological knowledge from query logs. In *Proceedings of the 16th international conference on World Wide Web, WWW '07*, pages 1223–1224, New York, NY, USA. ACM.
- Xiaoxin Yin and Sarthak Shah. 2010. Building taxonomy of web search intents for name entity queries. In *WWW*, pages 1001–1010.
- Z. Zhang and O. Nasraoui. 2006. Mining search engine query logs for query recommendation. In *WWW*, pages 1039–1040.

# Cross-Lingual Mixture Model for Sentiment Classification

Xinfan Meng<sup>‡</sup> \* Furu Wei<sup>†</sup> Xiaohua Liu<sup>†</sup> Ming Zhou<sup>†</sup> Ge Xu<sup>‡</sup> Houfeng Wang<sup>‡</sup>

<sup>‡</sup>MOE Key Lab of Computational Linguistics, Peking University

<sup>†</sup>Microsoft Research Asia

<sup>‡</sup>{mxf, xuge, wanghf}@pku.edu.cn

<sup>†</sup>{fuwei,xiaoliu,mingzhou}@microsoft.com

## Abstract

The amount of labeled sentiment data in English is much larger than that in other languages. Such a disproportion arouse interest in cross-lingual sentiment classification, which aims to conduct sentiment classification in the target language (e.g. Chinese) using labeled data in the source language (e.g. English). Most existing work relies on machine translation engines to directly adapt labeled data from the source language to the target language. This approach suffers from the limited coverage of vocabulary in the machine translation results. In this paper, we propose a generative cross-lingual mixture model (CLMM) to leverage unlabeled bilingual parallel data. By fitting parameters to maximize the likelihood of the bilingual parallel data, the proposed model learns previously unseen sentiment words from the large bilingual parallel data and improves vocabulary coverage significantly. Experiments on multiple data sets show that CLMM is consistently effective in two settings: (1) labeled data in the target language are unavailable; and (2) labeled data in the target language are also available.

## 1 Introduction

Sentiment Analysis (also known as opinion mining), which aims to extract the sentiment information from text, has attracted extensive attention in recent years. Sentiment classification, the task of determining the sentiment orientation (positive, negative or neutral) of text, has been the most extensively studied task in sentiment analysis. There is

already a large amount of work on sentiment classification of text in various genres and in many languages. For example, Pang et al. (2002) focus on sentiment classification of movie reviews in English, and Zagibalov and Carroll (2008) study the problem of classifying product reviews in Chinese. During the past few years, NTCIR<sup>1</sup> organized several pilot tasks for sentiment classification of news articles written in English, Chinese and Japanese (Seki et al., 2007; Seki et al., 2008).

For English sentiment classification, there are several labeled corpora available (Hu and Liu, 2004; Pang et al., 2002; Wiebe et al., 2005). However, labeled resources in other languages are often insufficient or even unavailable. Therefore, it is desirable to use the English labeled data to improve sentiment classification of documents in other languages. One direct approach to leveraging the labeled data in English is to use machine translation engines as a *black box* to translate the labeled data from English to the target language (e.g. Chinese), and then using the translated training data directly for the development of the sentiment classifier in the target language (Wan, 2009; Pan et al., 2011).

Although the machine-translation-based methods are intuitive, they have certain limitations. First, the vocabulary covered by the translated labeled data is limited, hence many sentiment indicative words can not be learned from the translated labeled data. Duh et al. (2011) report low overlapping between vocabulary of natural English documents and the vocabulary of documents translated to English from Japanese, and the experiments of Duh

\*Contribution during internship at Microsoft Research Asia.

<sup>1</sup><http://research.nii.ac.jp/ntcir/index-en.html>



et al. (2011) show that vocabulary coverage has a strong correlation with sentiment classification accuracy. Second, machine translation may change the sentiment polarity of the original text. For example, the negative English sentence “It is too good to be true” is translated to a positive sentence in Chinese “这是好得是真实的” by Google Translate (<http://translate.google.com/>), which literally means “It is good and true”.

In this paper we propose a cross-lingual mixture model (**CLMM**) for cross-lingual sentiment classification. Instead of relying on the unreliable machine translated labeled data, CLMM leverages bilingual parallel data to bridge the language gap between the source language and the target language. CLMM is a generative model that treats the source language and target language words in parallel data as generated *simultaneously* by a set of mixture components. By “synchronizing” the generation of words in the source language and the target language in a parallel corpus, the proposed model can (1) improve vocabulary coverage by learning sentiment words from the unlabeled parallel corpus; (2) transfer polarity label information between the source language and target language using a parallel corpus. Besides, CLMM can improve the accuracy of cross-lingual sentiment classification consistently regardless of whether labeled data in the target language are present or not. We evaluate the model on sentiment classification of Chinese using English labeled data. The experiment results show that CLMM yields 71% in accuracy when no Chinese labeled data are used, which significantly improves Chinese sentiment classification and is superior to the SVM and co-training based methods. When Chinese labeled data are employed, CLMM yields 83% in accuracy, which is remarkably better than the SVM and achieve state-of-the-art performance.

This paper makes two contributions: (1) we propose a model to effectively leverage large bilingual parallel data for improving vocabulary coverage; and (2) the proposed model is applicable in both settings of cross-lingual sentiment classification, irrespective of the availability of labeled data in the target language.

The paper is organized as follows. We review related work in Section 2, and present the cross-lingual mixture model in Section 3. Then we present the ex-

perimental studies in Section 4, and finally conclude the paper and outline the future plan in Section 5.

## 2 Related Work

In this section, we present a brief review of the related work on monolingual sentiment classification and cross-lingual sentiment classification.

### 2.1 Sentiment Classification

Early work of sentiment classification focuses on English product reviews or movie reviews (Pang et al., 2002; Turney, 2002; Hu and Liu, 2004). Since then, sentiment classification has been investigated in various domains and different languages (Zagiblov and Carroll, 2008; Seki et al., 2007; Seki et al., 2008; Davidov et al., 2010). There exist two main approaches to extracting sentiment orientation automatically. The Dictionary-based approach (Turney, 2002; Taboada et al., 2011) aims to aggregate the sentiment orientation of a sentence (or document) from the sentiment orientations of words or phrases found in the sentence (or document), while the corpus-based approach (Pang et al., 2002) treats the sentiment orientation detection as a conventional classification task and focuses on building classifier from a set of sentences (or documents) labeled with sentiment orientations.

Dictionary-based methods involve in creating or using sentiment lexicons. Turney (2002) derives sentiment scores for phrases by measuring the mutual information between the given phrase and the words “excellent” and “poor”, and then uses the average scores of the phrases in a document as the sentiment of the document. Corpus-based methods are often built upon machine learning models. Pang et al. (2002) compare the performance of three commonly used machine learning models (Naive Bayes, Maximum Entropy and SVM). Gamon (2004) shows that introducing deeper linguistic features into SVM can help to improve the performance. The interested readers are referred to (Pang and Lee, 2008) for a comprehensive review of sentiment classification.

### 2.2 Cross-Lingual Sentiment Classification

Cross-lingual sentiment classification, which aims to conduct sentiment classification in the target language (e.g. Chinese) with labeled data in the source

language (e.g. English), has been extensively studied in the very recent years. The basic idea is to explore the abundant labeled sentiment data in source language to alleviate the shortage of labeled data in the target language.

Most existing work relies on machine translation engines to directly adapt labeled data from the source language to target language. Wan (2009) proposes to use ensemble method to train better Chinese sentiment classification model on English labeled data and their Chinese translation. English Labeled data are first translated to Chinese, and then two SVM classifiers are trained on English and Chinese labeled data respectively. After that, co-training (Blum and Mitchell, 1998) approach is adopted to leverage Chinese unlabeled data and their English translation to improve the SVM classifier for Chinese sentiment classification. The same idea is used in (Wan, 2008), but the ensemble techniques used are various voting methods and the individual classifiers used are dictionary-based classifiers.

Instead of ensemble methods, Pan et al. (2011) use matrix factorization formulation. They extend Non-negative Matrix Tri-Factorization model (Li et al., 2009) to bilingual view setting. Their bilingual view is also constructed by using machine translation engines to translate original documents. Prettenhofer and Stein (2011) use machine translation engines in a different way. They generalize Structural Correspondence Learning (Blitzer et al., 2006) to multi-lingual setting. Instead of using machine translation engines to translate labeled text, the authors use it to construct the word translation oracle for pivot words translation.

Lu et al. (2011) focus on the task of jointly improving the performance of sentiment classification on two languages (e.g. English and Chinese). The authors use an unlabeled parallel corpus instead of machine translation engines. They assume parallel sentences in the corpus should have the same sentiment polarity. Besides, they assume labeled data in both language are available. They propose a method of training two classifiers based on maximum entropy formulation to maximize their prediction agreement on the parallel corpus. However, this method requires labeled data in *both* the source language and the target language, which are not always readily available.

### 3 Cross-Lingual Mixture Model for Sentiment Classification

In this section we present the cross-lingual mixture model (CLMM) for sentiment classification. We first formalize the task of cross-lingual sentiment classification. Then we describe the CLMM model and present the parameter estimation algorithm for CLMM.

#### 3.1 Cross-lingual Sentiment Classification

Formally, the task we are concerned about is to develop a sentiment classifier for the target language  $T$  (e.g. Chinese), given labeled sentiment data  $D_S$  in the source language  $S$  (e.g. English), unlabeled parallel corpus  $U$  of the source language and the target language, and *optional* labeled data  $D_T$  in target language  $T$ . Aligning with previous work (Wan, 2008; Wan, 2009), we only consider binary sentiment classification scheme (positive or negative) in this paper, but the proposed method can be used in other classification schemes with minor modifications.

#### 3.2 The Cross-Lingual Mixture Model

The basic idea underlying CLMM is to enlarge the vocabulary by learning sentiment words from the parallel corpus. CLMM defines an intuitive generation process as follows. Suppose we are going to generate a positive or negative Chinese sentence, we have two ways of generating words. The first way is to *directly* generate a Chinese word according to the polarity of the sentence. The other way is to first generate an English word with the same polarity and meaning, and then *translate* it to a Chinese word. More formally, CLMM defines a generative mixture model for generating a parallel corpus. The *unobserved* polarities of the unlabeled parallel corpus are modeled as hidden variables, and the *observed* words in parallel corpus are modeled as generated by a set of words generation distributions conditioned on the hidden variables. Given a parallel corpus, we fit CLMM model by maximizing the likelihood of generating this parallel corpus. By maximizing the likelihood, CLMM can estimate words generation probabilities for words unseen in the labeled data but present in the parallel corpus, hence expand the vocabulary. In addition, CLMM can utilize words in both the source language and target language for de-

terminating polarity classes of the parallel sentences.

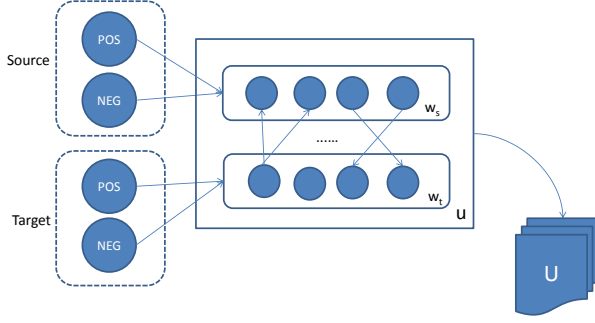


Figure 1: The generation process of the cross-lingual mixture model

Figure 1 illustrates the detailed process of generating words in the source language and target language respectively for the parallel corpus  $U$ , from the four mixture components in CLMM. Particularly, for each pair of parallel sentences  $u_i \in U$ , we generate the words as follows.

1. **Document class generation:** Generating the polarity class.
  - (a) Generating a polarity class  $c_s$  from a Bernoulli distribution  $P_s(C)$ .
  - (b) Generating a polarity class  $c_t$  from a Bernoulli distribution  $P_t(C)$
2. **Words generation:** Generating the words
  - (a) Generating source language words  $w_s$  from a Multinomial distribution  $P(w_s|c_s)$
  - (b) Generating target language words  $w_t$  from a Multinomial distribution  $P(w_t|c_t)$
3. **Words projection:** Projecting the words onto the other language
  - (a) Projecting the source language words  $w_s$  to target language words  $w_t$  by word projection probability  $P(w_t|w_s)$
  - (b) Projecting the target language words  $w_t$  to source language words  $w_s$  by word projection probability  $P(w_s|w_t)$

CLMM finds parameters by using MLE (Maximum Likelihood Estimation). The parameters to be estimated include conditional probabilities of word to class,  $P(w_s|c)$  and  $P(w_t|c)$ , and word projection

probabilities,  $P(w_s|w_t)$  and  $P(w_t|w_s)$ . We will describe the log-likelihood function and then show how to estimate the parameters in subsection 3.3. The obtained word-class conditional probability  $P(w_t|c)$  can then be used to classify text in the target languages using Bayes Theorem and the Naive Bayes independence assumption.

Formally, we have the following log-likelihood function for a parallel corpus  $U^2$ .

$$L(\theta|U) = \sum_{i=1}^{|U_s|} \sum_{j=1}^{|C|} \sum_{s=1}^{|V_s|} [N_{si} \log (P(w_s|c_j) + P(w_s|w_t)P(w_t|c_j))] + \sum_{i=1}^{|U_t|} \sum_{j=1}^{|C|} \sum_{t=1}^{|V_t|} [N_{ti} \log (P(w_t|c_j) + P(w_t|w_s)P(w_s|c_j))] \quad (1)$$

where  $\theta$  is the model parameters;  $N_{si}$  ( $N_{ti}$ ) is the occurrences of the word  $w_s$  ( $w_t$ ) in document  $d_i$ ;  $|D_s|$  is the number of documents;  $|C|$  is the number of class labels;  $V_s$  and  $V_t$  are the vocabulary in the source language and the vocabulary in the target language.  $|U_s|$  and  $|U_t|$  are the number of unlabeled sentences in the source language and target language.

Meanwhile, we have the following log-likelihood function for labeled data in the source language  $D_s$ .

$$L(\theta|D_s) = \sum_{i=1}^{|D_s|} \sum_{j=1}^{|C|} \sum_{s=1}^{|V_s|} N_{si} \log P(w_s|c_j) \delta_{ij} \quad (2)$$

where  $\delta_{ij} = 1$  if the label of  $d_i$  is  $c_j$ , and 0 otherwise.

In addition, when labeled data in the target language is available, we have the following log-likelihood function.

$$L(\theta|D_t) = \sum_{i=1}^{|D_t|} \sum_{j=1}^{|C|} \sum_{t=1}^{|V_t|} N_{ti} \log P(w_t|c_j) \delta_{ij} \quad (3)$$

Combining the above three likelihood functions together, we have the following likelihood function.

$$L(\theta|D_t, D_s, U) = L(\theta|U) + L(\theta|D_s) + L(\theta|D_t) \quad (4)$$

Note that the third term on the right hand side ( $L(\theta|D_t)$ ) is optional.

<sup>2</sup>For simplicity, we assume the prior distribution  $P(C)$  is uniform and drop it from the formulas.

### 3.3 Parameter Estimation

Instead of estimating word projection probability ( $P(w_s|w_t)$  and  $P(w_t|w_s)$ ) and conditional probability of word to class ( $P(w_t|c)$  and  $P(w_s|c)$ ) simultaneously in the training procedure, we estimate them separately since the word projection probability stays invariant when estimating other parameters. We estimate word projection probability using word alignment probability generated by the Berkeley aligner (Liang et al., 2006). The word alignment probabilities serves two purposes. First, they connect the corresponding words between the source language and the target language. Second, they adjust the strength of influences between the corresponding words. Figure 2 gives an example of word alignment probability. As is shown, the three words “tour de force” altogether express a positive meaning, while in Chinese the same meaning is expressed with only one word “杰作” (masterpiece). CLMM use word alignment probability to decrease the influences from “杰作” (masterpiece) to “tour”, “de” and “force” individually, using the word projection probability (i.e. word alignment probability), which is 0.3 in this case.

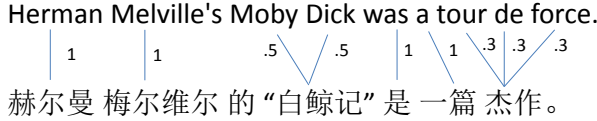


Figure 2: Word Alignment Probability

We use Expectation-Maximization (EM) algorithm (Dempster et al., 1977) to estimate the conditional probability of word  $w_s$  and  $w_t$  given class  $c$ ,  $P(w_s|c)$  and  $P(w_t|c)$  respectively. We derive the equations for EM algorithm, using notations similar to (Nigam et al., 2000).

In the E-step, the distribution of hidden variables (i.e. class label for unlabeled parallel sentences) is computed according to the following equations.

$$P(c_j|u_{si}) = Z(c_{u_{si}} = c_j) = \frac{\prod_{w_s \in u_{si}} [P(w_s|c_j) + \sum_{P(w_s|w_t) > 0} P(w_s|w_t)P(w_t|c_j)]}{\sum_{c_j} \prod_{w_s \in u_{si}} [P(w_s|c_j) + \sum_{P(w_s|w_t) > 0} P(w_s|w_t)P(w_t|c_j)]} \quad (5)$$

$$P(c_j|u_{ti}) = Z(c_{u_{ti}} = c_j) = \frac{\prod_{w_t \in u_{ti}} [P(w_t|c_j) + \sum_{P(w_t|w_s) > 0} P(w_t|w_s)P(w_s|c_j)]}{\sum_{c_j} \prod_{w_t \in u_{ti}} [P(w_t|c_j) + \sum_{P(w_t|w_s) > 0} P(w_t|w_s)P(w_s|c_j)]} \quad (6)$$

where  $Z(c_{u_{si}} = c_j)$  ( $Z(c_{u_{ti}} = c_j)$ ) is the probability of the source (target) language sentence  $u_{si}$  ( $u_{ti}$ ) in the  $i$ -th pair of sentences  $u_i$  having class label  $c_j$ .

In the M-step, the parameters are computed by the following equations.

$$P(w_s|c_j) = \frac{1 + \sum_{i=1}^{|D_s|} \Lambda_s(i) N_{si} P(c_j|d_i)}{|V| + \sum_{s=1}^{|V_s|} \Lambda(i) N_{si} P(c_j|d_i)} \quad (7)$$

$$P(w_t|c_j) = \frac{1 + \sum_{i=1}^{|D_t|} \Lambda_t(i) N_{ti} P(c_j|d_i)}{|V| + \sum_{t=1}^{|V_t|} \Lambda(i) N_{ti} P(c_j|d_i)} \quad (8)$$

where  $\Lambda_s(i)$  and  $\Lambda_t(i)$  are weighting factor to control the influence of the unlabeled data. We set  $\lambda_s(i)$  ( $\lambda_t(i)$ ) to  $\lambda_s$  ( $\lambda_t$ ) when  $d_i$  belongs to unlabeled data, 1 otherwise. When  $d_i$  belongs to labeled data,  $P(c_j|d_i)$  is 1 when its label is  $c_j$  and 0 otherwise. When  $d_i$  belongs to unlabeled data,  $P(c_j|d_i)$  is computed according to Equation 5 or 6.

## 4 Experiment

### 4.1 Experiment Setup and Data Sets

**Experiment setup:** We conduct experiments on two common cross-lingual sentiment classification settings. In the first setting, no labeled data in the target language are available. This setting has realistic significance, since in some situations we need to quickly develop a sentiment classifier for languages that we do not have labeled data in hand. In this case, we classify text in the target language using only labeled data in the source language. In the second setting, labeled data in the target language are also available. In this case, a more reasonable strategy is to make full use of both labeled data in the source language and target language to develop the sentiment classifier for the target language. In our experiments, we consider English as the source language and Chinese as the target language.

**Data sets:** For Chinese sentiment classification, we use the same data set described in (Lu et al., 2011). The labeled data sets consist of two English data sets and one Chinese data set. The English data set is from the Multi-Perspective Question Answering (MPQA) corpus (Wiebe et al., 2005) and the NTCIR Opinion Analysis Pilot Task data set (Seki et al., 2008; Seki et al., 2007). The Chinese data set also comes from the NTCIR Opinion Analysis Pilot Task data set. The unlabeled parallel sentences

are selected from ISI Chinese-English parallel corpus (Munteanu and Marcu, 2005). Following the description in (Lu et al., 2011), we remove neutral sentences and keep only high confident positive and negative sentences as predicted by a maximum entropy classifier trained on the labeled data. Table 1 shows the statistics for the data sets used in the experiments. We conduct experiments on two data settings: (1) MPQA + NTCIR-CH and (2) NTCIR-EN + NTCIR-CH.

	MPQA	NTCIR-EN	NTCIR-CH
Positive	1,471(30%)	528 (30%)	2,378 (55%)
Negative	3,487(70%)	1,209(70%)	1,916(44%)
Total	4,958	1,737	4,294

Table 1: Statistics about the Data

CLMM includes two hyper-parameters ( $\lambda_s$  and  $\lambda_t$ ) controlling the contribution of unlabeled parallel data. Larger weights indicate larger influence from the unlabeled data. We set the hyper-parameters by conducting cross validations on the labeled data. When Chinese labeled data are unavailable, we set  $\lambda_t$  to 1 and  $\lambda_s$  to 0.1, since no Chinese labeled data are used and the contribution of target language to the source language is limited. When Chinese labeled data are available, we set  $\lambda_s$  and  $\lambda_t$  to 0.2.

To prevent long sentences from dominating the parameter estimation, we preprocess the data set by normalizing the length of all sentences to the same constant (Nigam et al., 2000), the average length of the sentences.

## 4.2 Baseline Methods

For the purpose of comparison, we implement the following baseline methods.

*MT-SVM*: We translate the English labeled data to Chinese using Google Translate and use the translation results to train the SVM classifier for Chinese.

*SVM*: We train a SVM classifier on the Chinese labeled data.

*MT-Cotrain*: This is the co-training based approach described in (Wan, 2009). We summarize the main steps as follows. First, two monolingual SVM classifiers are trained on English labeled data and Chinese data *translated* from English labeled data. Second, the two classifiers make prediction on Chinese unlabeled data and their English translation,

respectively. Third, the 100 most confidently predicted English and Chinese sentences are added to the training set and the two monolingual SVM classifiers are re-trained on the expanded training set. The second and the third steps are repeated for 100 times to obtain the final classifiers.

*Para-Cotrain*: The training process is the same as MT-Cotrain. However, we use a different set of English unlabeled sentences. Instead of using the corresponding machine translation of Chinese unlabeled sentences, we use the parallel English sentences of the Chinese unlabeled sentences.

*Joint-Train*: This is the state-of-the-art method described in (Lu et al., 2011). This model use English labeled data and Chinese labeled data to obtain initial parameters for two maximum entropy classifiers (for English documents and Chinese documents), and then conduct EM-iterations to update the parameters to gradually improve the agreement of the two monolingual classifiers on the unlabeled parallel sentences.

## 4.3 Classification Using Only English Labeled Data

The first set of experiments are conducted on using only English labeled data to create the sentiment classifier for Chinese. This is a challenging task, since we do not use any Chinese labeled data. And MPQA and NTCIR data sets are compiled by different groups using different annotation guidelines.

Method	NTCIR-EN NTCIR-CH	MPQA-EN NTCIR-CH
MT-SVM	62.34	54.33
SVM	N/A	N/A
MT-Cotrain	65.13	59.11
Para-Cotrain	67.21	60.71
Joint-Train	N/A	N/A
CLMM	<b>70.96</b>	<b>71.52</b>

Table 2: Classification Accuracy Using Only English Labeled Data

Table 2 shows the accuracy of the baseline systems as well as the proposed model (CLMM). As is shown, sentiment classification does not benefit much from the direct machine translation. For NTCIR-EN+NTCIR-CH, the accuracy of MT-SVM

is only 62.34%. For MPQA-EN+NTCIR-CH, the accuracy is 54.33%, even lower than a trivial method, which achieves 55.4% by predicting all sentences to be positive. The underlying reason is that the vocabulary coverage in machine translated data is low, therefore the classifier learned from the labeled data is unable to generalize well on the test data. Meanwhile, the accuracy of MT-SVM on NTCIR-EN+NTCIR-CH data set is much better than that on MPQA+NTCIR-CH data set. That is because NTCIR-EN and NTCIR-CH cover similar topics. The other two methods using machine translated data, MT-Cotrain and Para-Cotrain also do not perform very well. This result is reasonable, because the initial Chinese classifier trained on machine translated data (MT-SVM) is relatively weak. We also observe that using a parallel corpus instead of machine translations can improve classification accuracy. It should be noted that we do not have the result for Joint-Train model in this setting, since it requires both English labeled data and Chinese labeled data.

#### 4.4 Classification Using English and Chinese Labeled Data

The second set of experiments are conducted on using both English labeled data and Chinese labeled data to develop the Chinese sentiment classifier. We conduct 5-fold cross validations on Chinese labeled data. We use the same baseline methods as described in Section 4.2, but we use *natural* Chinese sentences instead of *translated* Chinese sentences as labeled data in MT-Cotrain and Para-Cotrain. Table 3 shows the accuracy of baseline systems as well as CLMM.

Method	NTCIR-EN NTCIR-CH	MPQA-EN NTCIR-CH
MT-SVM	62.34	54.33
SVM	80.58	80.58
MT-Cotrain	82.28	80.93
Para-Cotrain	82.35	82.18
Joint-Train	83.11	83.42
CLMM	82.73	83.02

Table 3: Classification Accuracy Using English and Chinese Labeled Data

As is seen, SVM performs significantly better than MT-SVM. One reason is that we use natural Chi-

nese labeled data instead of translated Chinese labeled data. Another reason is that we use 5-fold cross validations in this setting, while the previous setting is an *open test* setting. In this setting, SVM is a strong baseline with 80.6% accuracy. Nevertheless, all three methods which leverage an unlabeled parallel corpus, namely Para-Cotrain, Joint-Train and CLMM, still show big improvements over the SVM baseline. Their results are comparable and all achieve state-of-the-art accuracy of about 83%, but in terms of training speed, CLMM is the fastest method (Table 4). Similar to the previous setting, We also have the same observation that using a parallel corpus is better than using translations.

Method	Iterations	Total Time
Para-Cotrain	100	6 hours
Joint-Train	10	55 seconds
CLMM	10	30 seconds

Table 4: Training Speed Comparison

#### 4.5 The Influence of Unlabeled Parallel Data

We investigate how the size of the unlabeled parallel data affects the sentiment classification in this subsection. We vary the number of sentences in the unlabeled parallel from 2,000 to 20,000. We use only English labeled data in this experiment, since this more directly reflects the effectiveness of each model in utilizing unlabeled parallel data. From Figure 3 and Figure 4, we can see that when more unlabeled parallel data are added, the accuracy of CLMM consistently improves. The performance of CLMM is remarkably superior than Para-Cotrain and MT-Cotrain. When we have 10,000 parallel sentences, the accuracy of CLMM on the two data sets quickly increases to 68.77% and 68.91%, respectively. By contrast, we observe that the performance of Para-Cotrain and MT-Cotrain is able to obtain accuracy improvement only after about 10,000 sentences are added. The reason is that the two methods use machine translated labeled data to create initial Chinese classifiers. As is depicted in Table 2, these classifiers are relatively weak. As a result, in the initial iterations of co-training based methods, the predictions made by the Chinese classifiers are inaccurate, and co-training based methods need to see more parallel

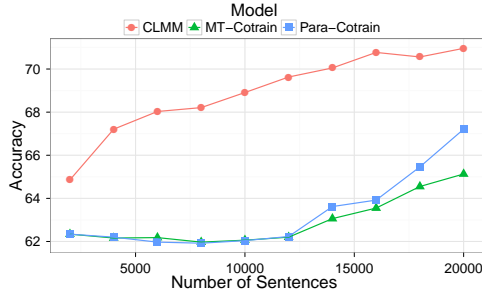


Figure 3: Accuracy with different size of unlabeled data for NTICR-EN+NTCIR-CH

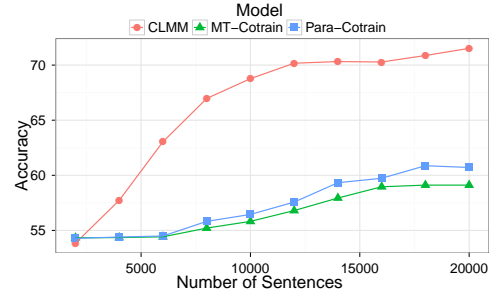


Figure 4: Accuracy with different size of unlabeled data for MPQA+NTCIR-CH

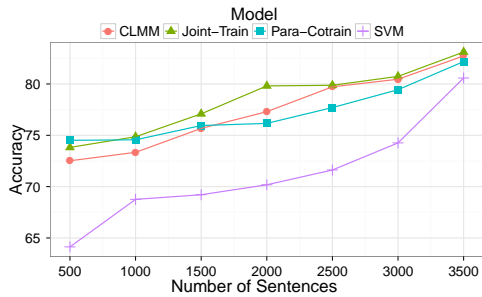


Figure 5: Accuracy with different size of labeled data for NTCIR-EN+NTCIR-CH

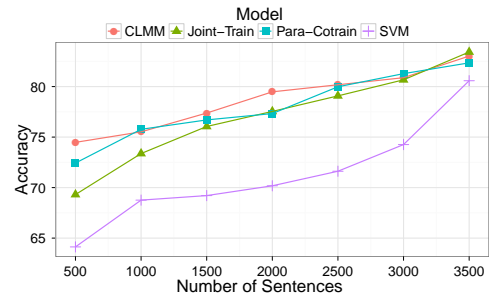


Figure 6: Accuracy with different size of labeled data for MPQA+NTCIR-CH

sentences to refine the initial classifiers.

#### 4.6 The Influence of Chinese Labeled Data

In this subsection, we investigate how the size of the Chinese labeled data affects the sentiment classification. As is shown in Figure 5 and Figure 6, when only 500 labeled sentences are used, CLMM is capable of achieving 72.52% and 74.48% in accuracy on the two data sets, obtaining 10% and 8% improvements over the SVM baseline, respectively. This indicates that our method leverages the unlabeled data effectively. When more sentences are used, CLMM consistently shows further improvement in accuracy. Para-Cotrain and Joint-Train show similar trends. When 3500 labeled sentences are used, SVM achieves 80.58%, a relatively high accuracy for sentiment classification. However, CLMM and the other two models can still gain improvements. This further demonstrates the advantages of expanding vocabulary using bilingual parallel data.

## 5 Conclusion and Future Work

In this paper, we propose a cross-lingual mixture model (CLMM) to tackle the problem of cross-lingual sentiment classification. This method has two advantages over the existing methods. First, the proposed model can learn previously unseen sentiment words from large unlabeled data, which are not covered by the limited vocabulary in machine translation of the labeled data. Second, CLMM can effectively utilize unlabeled parallel data regardless of whether labeled data in the target language are used or not. Extensive experiments suggest that CLMM consistently improve classification accuracy in both settings. In the future, we will work on leveraging parallel sentences and word alignments for other tasks in sentiment analysis, such as building multi-lingual sentiment lexicons.

**Acknowledgment** We thank Bin Lu and Lei Wang for their help. This research was partly supported by National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101) and National Natural Science Foundation of China (No.91024009, No.60973053)



## References

- John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, page 120–128.
- Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, page 92–100.
- Dmitry Davidov, Oren Tsur, and Ari Rappoport. 2010. Enhanced sentiment learning using twitter hashtags and smileys. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, page 241–249.
- Arthur Dempster, Nan Laird, and Donald Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, page 1–38.
- Kevin Duh, Akinori Fujino, and Masaaki Nagata. 2011. Is machine translation ripe for Cross-Lingual sentiment classification? In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, page 429–433, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Michael Gamon. 2004. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the 20th international conference on Computational Linguistics*, page 841.
- Mingqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, page 168–177.
- Tao Li, Yi Zhang, and Vikas Sindhwani. 2009. A non-negative matrix tri-factorization approach to sentiment classification with lexical prior knowledge. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, page 244–252, Suntec, Singapore, August. Association for Computational Linguistics.
- Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, page 104–111.
- Bin Lu, Chenhao Tan, Claire Cardie, and Benjamin K. Tsou. 2011. Joint bilingual sentiment classification with unlabeled parallel corpora. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies—Volume 1*, page 320–330.
- Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, 31(4):477–504.
- Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 39(2):103–134.
- Junfeng Pan, Gui-Rong Xue, Yong Yu, and Yang Wang. 2011. Cross-lingual sentiment classification via bi-view non-negative matrix tri-factorization. *Advances in Knowledge Discovery and Data Mining*, page 289–300.
- Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January.
- Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing—Volume 10*, page 79–86.
- Peter Prettenhofer and Benno Stein. 2011. Cross-lingual adaptation using structural correspondence learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(1):13.
- Yohei Seki, David Kirk Evans, Lun-Wei Ku, Hsin-Hsi Chen, Noriko Kando, and Chin-Yew Lin. 2007. Overview of opinion analysis pilot task at NTCIR-6. In *Proceedings of NTCIR-6 Workshop Meeting*, page 265–278.
- Yohei Seki, David Kirk Evans, Lun-Wei Ku, Le Sun, Hsin-Hsi Chen, Noriko Kando, and Chin-Yew Lin. 2008. Overview of multilingual opinion analysis task at NTCIR-7. In *Proc. of the Seventh NTCIR Workshop*.
- Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-Based methods for sentiment analysis. *Comput. Linguist.*, page to appear.
- Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, page 417–424.
- Xiaojun Wan. 2008. Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, page 553–561, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Xiaojun Wan. 2009. Co-training for cross-lingual sentiment classification. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and*



- the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, page 235–243.
- Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2):165–210.
- Taras Zagibalov and John Carroll. 2008. Automatic seed word selection for unsupervised sentiment classification of chinese text. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, page 1073–1080.

# Community Answer Summarization for Multi-Sentence Question with Group $L_1$ Regularization

Wen Chan<sup>†</sup>, Xiangdong Zhou<sup>†</sup>, Wei Wang<sup>†</sup>, Tat-Seng Chua<sup>‡</sup>

<sup>†</sup>School of Computer Science, Fudan University, Shanghai, 200433, China

{11110240007, xdzhou, weiwang1}@fudan.edu.cn

<sup>‡</sup>School of Computing, National University of Singapore

chuats@nus.edu.sg

## Abstract

We present a novel answer summarization method for community Question Answering services (cQAs) to address the problem of “incomplete answer”, i.e., the “best answer” of a complex multi-sentence question misses valuable information that is contained in other answers. In order to automatically generate a novel and non-redundant community answer summary, we segment the complex original multi-sentence question into several sub questions and then propose a general Conditional Random Field (CRF) based answer summary method with group  $L_1$  regularization. Various textual and non-textual QA features are explored. Specifically, we explore four different types of contextual factors, namely, the information novelty and non-redundancy modeling for local and non-local sentence interactions under question segmentation. To further unleash the potential of the abundant cQA features, we introduce the group  $L_1$  regularization for feature learning. Experimental results on a Yahoo! Answers dataset show that our proposed method significantly outperforms state-of-the-art methods on cQA summarization task.

## 1 Introduction

Community Question and Answering services (cQAs) have become valuable resources for users to pose questions of their interests and share their knowledge by providing answers to questions. They perform much better than the traditional frequently asked questions (FAQ) systems (Jijkoun and Rijke, 2005; Riezler et al., 2007) which are just based

on natural language processing and information retrieving technologies due to the need for human intelligence in user generated contents (Gyongyi et al., 2007). In cQAs such as Yahoo! Answers, a resolved question often gets more than one answers and a “best answer” will be chosen by the asker or voted by other community participants. This {question, best answer} pair is then stored and indexed for further uses such as question retrieval. It performs very well in simple factoid QA settings, where the answers to factoid questions often relate to a single named entity like a person, time or location. However, when it comes to the more sophisticated multi-sentence questions, it would suffer from the problem of “incomplete answer”. That is, such question often comprises several sub questions in specific contexts and the asker wishes to get elaborated answers for as many aspects of the question as possible. In which case, the single best answer that covers just one or few aspects may not be a good choice (Liu et al., 2008; Takechi et al., 2007). Since “everyone knows something” (Adamic et al., 2008), the use of a single best answer often misses valuable human generated information contained in other answers.

In an early literature, Liu et al. (2008) reported that no more than 48% of the 400 best answers were indeed the unique best answers in 4 most popular Yahoo! Answers categories. Table 1 shows an example of the “incomplete answer” problem from Yahoo! Answers<sup>1</sup>. The asker wishes to know why his teeth bleed and how to prevent it. However, the best answer only gives information on the reason of teeth

<sup>1</sup><http://answers.yahoo.com/question/index?qid=20100610161858AAmAGrV>

bleeding. It is clear that some valuable information about the reasons of gums bleeding and some solutions are presented in other answers.

<p><b>Question</b></p> <p>Why do teeth bleed at night and how do you prevent/stop it? This morning I woke up with blood caked between my two front teeth. This is the third morning in a row that it has happened. I brush and floss regularly, and I also eat a balanced, healthy diet. Why is this happening and how do I stop it?</p>
<p><b>Best Answer - Chosen by Asker</b></p> <p>Periodontal disease is a possibility, gingivitis, or some gum infection. Teeth don't bleed; gums bleed.</p>
<p><b>Other Answers</b></p> <p>Vitamin C deficiency!</p> <p>Ever heard of a dentist? Not all the problems in life are solved on the Internet.</p> <p>You could be brushing or flossing too hard. Try a brush with softer bristles or brushing/flossing lighter and slower. If this doesn't solve your problem, try seeing a dentist or doctor. Gums that bleed could be a sign of a more serious issue like leukemia, an infection, gum disease, a blood disorder, or a vitamin deficiency.</p> <p>wash your mouth with warm water and salt, it will help to strengthen your gum and teeth, also salt avoid infection. You probably have weak gums, so just try to follow the advice, it works in many cases of oral problems.</p>

Table 1: An example of question with incomplete answer problem from Yahoo! Answers. The “best answer” seems to miss valuable information and will not be ideal for re-use when similar question is asked again.

In general, as noted in (Jurafsky and Martin, 2009), most interesting questions are not factoid questions. User’s needs require longer, more informative answers than a single phrase. In fact, it is often the case, that a complex multi-sentence question could be answered from multiple aspects by different people focusing on different sub questions. Therefore we address the incomplete answer problem by developing a novel summarization technique taking different sub questions and contexts into consideration. Specifically we want to learn a concise summary from a set of corresponding answers as supplement or replacement to the “best answer”.

We tackle the answer summary task as a sequential labeling process under the general Conditional Random Fields (CRF) framework: every answer sentence in the question thread is labeled as a *summary* sentence or *non-summary* sentence, and we concatenate the sentences with *summary* label to form the final summarized answer. The contribution of this paper is two-fold:

First, we present a general CRF based framework

and incorporate four different contextual factors based on question segmentation to model the local and non-local semantic sentence interactions to address the problem of redundancy and information novelty. Various textual and non-textual question answering features are exploited in the work.

Second, we propose a group  $L_1$ -regularization approach in the CRF model for automatic optimal feature learning to unleash the potential of the features and enhance the performance of answer summarization.

We conduct experiments on a Yahoo! Answers dataset. The experimental results show that the proposed model improve the performance significantly (in terms of precision, recall and F1 measures) as well as the ROUGE-1, ROUGE-2 and ROUGE-L measures as compared to the state-of-the-art methods, such as Support Vector Machines (SVM), Logistic Regression (LR) and Linear CRF (LCRF) (Shen et al., 2007).

The rest of the paper is arranged as follows: Section 2 presents some definitions and a brief review of related research. In Section 3, we propose the summarization framework and then in Section 4 and 5 we detail the experimental setups and results respectively. We conclude the paper in Section 6.

## 2 Definitions and Related Work

### 2.1 Definitions

In this subsection we define some concepts that would be helpful to clarify our problems. First we define a *complex multi-sentence* question as a question with the following properties:

**Definition:** A *complex multi-sentence question* is one that contains multiple sub-questions.

In the cQAs scenario a question often consists of one or more main question sentences accompany by some context sentences described by askers. We treat the original question and context as a whole single complex multi-sentence question and obtain the sub questions by question segmentation. We then define the *incomplete answer* problem as:

**Definition:** The *incomplete answer problem* is one where the best answer of a complex multi-sentence question is voted to be below certain star ratings or the average similarity between the best answer and all the sub questions is below some thresh-

olds.

We study the issues of similarity threshold and the minimal number of stars empirically in the experimental section and show that they are useful in identifying questions with the incomplete answer problem.

## 2.2 Related Work

There exist several attempts to alleviate the answer completeness problem in cQA. One of them is to segment the multi-sentence question into a set of sub-questions along with their contexts, then sequentially retrieve the sub questions one by one, and return similar questions and their best answers (Wang et al., 2010). This strategy works well in general, however, as the automatic question segmentation is imperfect and the matched similar questions are likely to be generated in different contextual situations, this strategy often could not combine multiple independent best answers of sub questions seamlessly and may introduce redundancy in final answer.

On general problem of cQA answer summarization, Liu et al.(2008) manually classified both questions and answers into different taxonomies and applied clustering algorithms for answer summarization. They utilized textual features for open and opinion type questions. Through exploiting metadata, Tomasoni and Huang(2010) introduced four characteristics (constraints) of summarized answer and combined them in an additional model as well as a multiplicative model. In order to leverage context, Yang et al.(2011) employed a dual wing factor graph to mutually enhance the performance of social document summarization with user generated content like tweets. Wang et al. (2011) learned online discussion structures such as the replying relationship by using the general CRFs and presented a detailed description of their feature designs for sites and edges embedded in discussion thread structures. However there is no previous work that explores the complex multi-sentence question segmentation and its contextual modeling for community answer summarization.

Some other works examined the evaluation of the quality of features for answers extracted from cQA services (Jeon et al., 2006; Hong and Davison , 2009; Shah et al., 2010). In the work of Shah et al.(2010), a large number of features extracted for

predicting asker-rated quality of answers was evaluated by using a logistic regression model. However, to the best of our knowledge, there is no work in evaluating the quality of features for community answer summarization. In our work we model the feature learning and evaluation problem as a group  $L_1$  regularization problem (Schmidt , 2010) on different feature groups.

## 3 The Summarization Framework

### 3.1 Conditional Random Fields

We utilize the probabilistic graphical model to solve the answer summarization task, Figure 1 gives some illustrations, in which the sites correspond to the sentences and the edges are utilized to model the interactions between sentences. Specifically, let  $\mathbf{x}$  be the sentence sequence to all answers within a question thread, and  $\mathbf{y}$  be the corresponding label sequence. Every component  $y_i$  of  $\mathbf{y}$  has a binary value, with +1 for the summary sentence and -1 otherwise. Then under CRF (Lafferty et al., 2001), the conditional probability of  $\mathbf{y}$  given  $\mathbf{x}$  obeys the following distribution:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_{v \in V, l} \mu_l g_l(v, \mathbf{y}|_v, \mathbf{x}) + \sum_{e \in E, k} \lambda_k f_k(e, \mathbf{y}|_e, \mathbf{x})\right), \quad (1)$$

where  $Z(\mathbf{x})$  is the normalization constant called partition function,  $g_l$  denotes the cQA feature function of site  $l$ ,  $f_k$  denotes the function of edge  $k$  (modeling the interactions between sentences),  $\mu$  and  $\lambda$  are respectively the weights of function of sites and edges, and  $\mathbf{y}|_t$  denotes the components of  $\mathbf{y}$  related to site (edge)  $t$ .

### 3.2 cQA Features and Contextual Modeling

In this section, we give a detailed description of the different sentence-level cQA features and the contextual modeling between sentences used in our model for answer summarization.

#### Sentence-level Features

Different from the conventional multi-document summarization in which only the textual features are utilized, we also explore a number of non-textual author related features (Shah et al., 2010) in cQAs.

**The textual features used are:**

1. **Sentence Length:** The length of the sentence in the answers with the stop words removed. It seems that a long sentence may contain more information.
2. **Position:** The sentence's position within the answer. If a sentence is at the beginning or at the end of one answer, it might be a generation or viewpoint sentence and will be given higher weight in the summarization task.
3. **Answer Length:** The length of the answer to which the sentence belonged, again with the stop words removed.
4. **Stopwords Rate:** The rate of stop words in the sentence. If a sentence contains too many stop words, it is more likely a spam or chitchat sentence rather than an informative one.
5. **Uppercase Rate:** The rate of uppercase words. Uppercase words are often people's name, address or other name entities interested by askers.
6. **Has Link** Whether the sentence contains a hyperlink or not. The link often points to a more detailed information source.
7. **Similarity to Question:** Semantic similarity to the question and question context. It imports the semantic information relevance to the question and question context.

**The non-textual features used include:**

8. **Best Answer Star:** The stars of the best answer received by the askers or voters.
9. **Thumbs Up:** The number of thumbs-ups the answer which contains the sentence receives. Users are often used to support one answer by giving a thumbs up after reading some relevant or interesting information for their intentions.
10. **Author Level:** The level of stars the author who gives the answer sentence acquires. The higher the star level, the more authoritative the asker is.
11. **Best Answer Rate:** Rate of answers annotated as the best answer the author who gives the answer sentence receives.
12. **Total Answer Number:** The number of total answers by the author who gives the answer sentence. The more answers one gives, the more experience he or she acquires.
13. **Total Points:** The total points that the author who gives the answer sentence receives.

The previous literature (Shah et al., 2010) hinted that some cQA features, such as *Sentence Length*, *Has Link* and *Best Answer Star*, may be more im-

portant than others. We also expect that some feature may be redundant when their most related features are given, e.g., the *Author Level* feature is positively related with the *Total Points* received by answerers, and *Stopwords Rate* is of little help when both *Sentence Length* (not including stop words) and *Uppercase Rate* are given. Therefore, to explore the optimal combination of these features, we propose a group  $L_1$  regularization term in the general CRF model (Section 3.3) for feature learning.

All features presented here can be extracted automatically from the Yahoo! Answers website. We normalize all these feature values to real numbers between 0 and 1 by dividing them by the corresponding maximal value of these features. These sentence-level features can be easily utilized in the CRF framework. For instance, if the rate of uppercase words is prominent or the position is close to the beginning or end of the answer, then the probability of the label +1 (summary sentence) should be boosted by assigning it with a large value.

### Contextual Modeling Under Question Segmentation

For cQAs summarization, the semantic interactions between different sentence sites are crucial, that is, some context co-occurrences should be encouraged and others should be penalized for requirements of information novelty and non-redundancy in the generated summary. Here we consider both local (sentences from the same answer) and global (sentences from different answers) settings. This give rise to four contextual factors that we will explore for modeling the pairwise semantic interactions based on question segmentation. In this paper, we utilize a simple but effective lightweight question segmentation method (Ding et al., 2008; Wang et al., 2010). It mainly involves the following two steps:

**Step 1.** Question sentence detection: every sentence in the original multi-sentence question is classified into *question sentence* and *non-question (context) sentence*. The *question mark* and *5WIH* features are applied.

**Step 2.** Context assignment: every context sentence is assigned to the most relevant question sentence. We compute the semantic similarity (Simpson and Crowe, 2005) between sentences or sub ques-

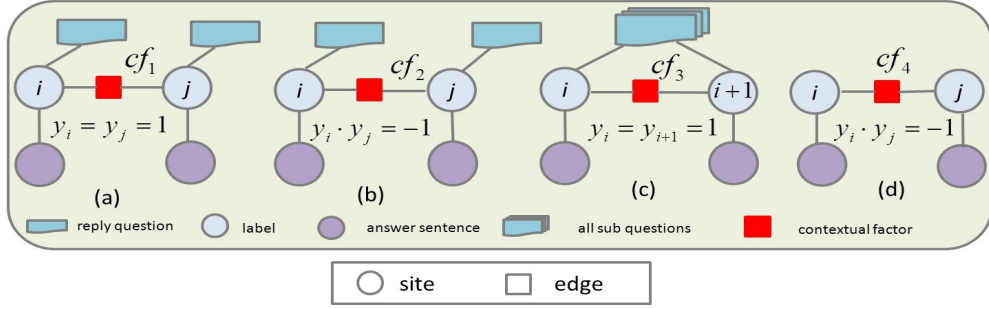


Figure 1: Four kinds of the contextual factors are considered for answer summarization in our general CRF based models.

tions as:

$$sim(x, y) = 2 \times \sum_{(w_1, w_2) \in M(x, y)} \frac{sim(w_1, w_2)}{|x| + |y|} \quad (2)$$

where  $M(x, y)$  denotes synset pairs matched in sentences  $x$  and  $y$ ; and the similarity between the two synsets  $w_1$  and  $w_2$  is computed to be inversely proportional to the length of the path in Wordnet.

One answer sentence may related to more than one sub questions to some extent. Thus, we define the *replied question*  $Qr_i$  as the sub question with the maximal similarity to sentence  $x_i$ :  $Qr_i = \operatorname{argmax}_{Q_j} sim(x_i, Q_j)$ . It is intuitive that different summary sentences aim at answering different sub questions. Therefore, we design the following two contextual factors based on the similarity of replied questions.

**Dissimilar Replied Question Factor:** Given two answer sentences  $x_i$ ,  $x_j$  and their corresponding replied questions  $Qr_i$ ,  $Qr_j$ . If the similarity<sup>2</sup> of  $Qr_i$  and  $Qr_j$  is below some threshold  $\tau_{lq}$ , it means that  $x_i$  and  $x_j$  will present different viewpoints to answer different sub questions. In this case, it is likely that  $x_i$  and  $x_j$  are both summary sentences; we ensure this by setting the contextual factor  $cf_1$  with a large value of  $\exp \nu$ , where  $\nu$  is a positive real constant often assigned to value 1; otherwise we set  $cf_1$  to  $\exp - \nu$  for penalization.

$$cf_1 = \begin{cases} \exp \nu, & y_i = y_j = 1 \\ \exp - \nu, & \text{otherwise} \end{cases}$$

**Similar Replied Question Factor:** Given two an-

<sup>2</sup>We use the semantic similarity of Equation 2 for all our similarity measurement in this paper.

swer sentences  $x_i$ ,  $x_j$  and their corresponding replied questions  $Qr_i$ ,  $Qr_j$ . If the similarity of  $Qr_i$  and  $Qr_j$  is above some upper threshold  $\tau_{uq}$ , this means that  $x_i$  and  $x_j$  are very similar and likely to provide similar viewpoint to answer similar questions. In this case, we want to select either  $x_i$  or  $x_j$  as answer. This is done by setting the contextual factor  $cf_2$  such that  $x_i$  and  $x_j$  have opposite labels,

$$cf_2 = \begin{cases} \exp \nu, & y_i * y_j = -1 \\ \exp - \nu, & \text{otherwise} \end{cases}$$

Assuming that sentence  $x_i$  is selected as a summary sentence, and its next local neighborhood sentence  $x_{i+1}$  by the same author is dissimilar to it but it is relevant to the original multi-sentence question, then it is reasonable to also pick  $x_{i+1}$  as a summary sentence because it may offer new viewpoints by the author. Meanwhile, other local and non-local sentences which are similar to it at above the upper threshold will probably not be selected as summary sentences as they offer similar viewpoint as discussed above. Therefore, we propose the following two kinds of contextual factors for selecting the answer sentences in the CRF model.

**Local Novelty Factor:** If the similarity of answer sentence  $x_i$  and  $x_{i+1}$  given by the same author is below a lower threshold  $\tau_{ls}$ , but their respective similarities to the sub questions both exceed an upper threshold  $\tau_{us}$ , then we will boost the probability of selecting both as summary sentences by setting:

$$cf_3 = \begin{cases} \exp \nu, & y_i = y_{i+1} = 1 \\ \exp - \nu, & \text{otherwise} \end{cases}$$

**Redundance Factor:** If the similarity of answer

sentence  $x_i$  and  $x_j$  is greater than the upper threshold  $\tau_{us}$ , then they are likely to be redundant and hence should be given opposite labels. This is done by setting:

$$cf_4 = \begin{cases} \exp \nu, & y_i * y_j = -1 \\ \exp -\nu, & \text{otherwise} \end{cases}$$

Figure 1 gives an illustration of these four contextual factors in our proposed general CRF based model. The parameter estimation and model inference are discussed in the following subsection.

### 3.3 Group $L_1$ Regularization for Feature Learning

In the context of cQA summarization task, some features are intuitively to be more important than others. As a result, we group the parameters in our CRF model with their related features<sup>3</sup> and introduce a group  $L_1$ -regularization term for selecting the most useful features from the least important ones, where the regularization term becomes,

$$R(\theta) = C \sum_{g=1}^G \|\vec{\theta}_g\|_2, \quad (3)$$

where  $C$  controls the penalty magnitude of the parameters,  $G$  is the number of feature groups and  $\vec{\theta}_g$  denotes the parameters corresponding to the particular group  $g$ . Notice that this penalty term is indeed a  $L(1, 2)$  regularization because in every particular group we normalize the parameters in  $L_2$  norm while the weight of a whole group is summed in  $L_1$  form.

Given a set of training data  $D = (x^{(i)}, y^{(i)})$ ,  $i = 1, \dots, N$ , the parameters  $\theta = (\mu_l, \lambda_k)$  of the general CRF with the group  $L_1$ -regularization are estimated in using a maximum log likelihood function  $L$  as:

$$L = \sum_{i=1}^N \log(p_\theta(y^{(i)}|x^{(i)})) - C \sum_{g=1}^G \|\vec{\theta}_g\|_2, \quad (4)$$

<sup>3</sup>We note that every sentence-level feature discussed in Section 3.2 presents a variety of instances (e.g., the sentence with longer or shorter length is the different instance), and we may call it sub-feature of the original sentence-level feature in the micro view. Every sub-feature has its corresponding weight in our CRF model. Whereas in a macro view, those related sub-features can be considered as a group.

where  $N$  denotes the total number of training samples. we compute the log-likelihood gradient component of  $\theta$  in the first term of Equation 4 as in usual CRFs. However, the second term of Equation 4 is non-differentiable when some special  $\|\vec{\theta}_g\|_2$  becomes exactly zero. To tackle this problem, an additional variable is added for each group (Schmidt, 2010); that is, by replacing each norm  $\|\vec{\theta}_g\|_2$  with the variable  $\alpha_g$ , subject to the constraint  $\alpha_g \geq \|\vec{\theta}_g\|_2$ , i.e.,

$$L = \sum_{i=1}^N \log(p_\theta(y^{(i)}|x^{(i)})) - C \sum_{g=1}^G \alpha_g, \quad (5)$$

subject to  $\alpha_g \geq \|\vec{\theta}_g\|_2, \forall g$ .

This formulation transforms the non-differentiable regularizer to a simple linear function and maximizing Equation 5 will lead to a solution to Equation 4 because it is a lower bound of the latter. Then, we add a sufficient small positive constant  $\varepsilon$  when computing the  $L_2$  norm (Lee et al., 2006), i.e.,  $\|\vec{\theta}_g\|_2 = \sqrt{\sum_{j=1}^{|g|} \theta_{gj}^2 + \varepsilon}$ , where  $|g|$  denotes the number of features in group  $g$ . To obtain the optimal value of parameter  $\theta$  from the training data, we use an efficient L-BFGS solver to solve the problem, and the first derivative of every feature  $j$  in group  $g$  is,

$$\frac{\delta L}{\delta \theta_{gj}} = \sum_{i=1}^N C_{gj}(y^{(i)}, x^{(i)}) - \sum_{i=1}^N \sum_y p(y|x^{(i)}) C_{gj}(y, x^{(i)}) - 2C \frac{\theta_{gj}}{\sqrt{\sum_{l=1}^{|g|} \theta_{gl}^2 + \varepsilon}} \quad (6)$$

where  $C_{gj}(y, x)$  denotes the count of feature  $j$  in group  $g$  of observation-label pair  $(x, y)$ . The first two terms of Equation 6 measure the difference between the empirical and the model expected values of feature  $j$  in group  $g$ , while the third term is the derivative of group  $L_1$  priors.

For inference, the labeling sequence can be obtained by maximizing the probability of  $y$  conditioned on  $x$ ,

$$y^* = \operatorname{argmax}_y p_\theta(y|x). \quad (7)$$

We use a modification of the Viterbi algorithm to perform inference of the CRF with non-local edges

previously used in (Galley, 2006). That is, we replace the edge connection  $z_t = (y_{t-2}, y_{t-1}, y_t)$  of order-2 Markov model by  $z_t = (y_{N_t}, y_{t-1}, y_t)$ , where  $y_{N_t}$  represents the label at the source of the non-local edge. Although it is an approximation of the exact inference, we will see that it works well for our answer summarization task in the experiments.

## 4 Experimental Setting

### 4.1 Dataset

To evaluate the performance of our CRF based answer summarization model, we conduct experiments on the Yahoo! Answers archives dataset. The Yahoo! *Webscope*<sup>TM</sup> Program<sup>4</sup> opens up a number of Yahoo! Answers datasets for interested academics in different categories. Our original dataset contains 1,300,559 questions and 2,770,896 answers in ten taxonomies from Yahoo! Answers. After filtering the questions which have less than 5 answers and some trivial factoid questions using the features by (Tomasoni and Huang, 2010), we reduce the dataset to 55,132 questions. From this sub-set, we next select the questions with incomplete answers as defined in Section 2.1. Specifically, we select the questions where the average similarity between the best answer and all sub questions is less than 0.6 or when the star rating of the best answer is less than 4. We obtain 7,784 questions after this step. To evaluate the effectiveness of this method, we randomly choose 400 questions in the filtered dataset and invite 10 graduate candidate students (not in NLP research field) to verify whether a question suffers from the incomplete answer problem. We divide the students into five groups of two each. We consider the questions as the “incomplete answer questions” only when they are judged by both members in a group to be the case. As a result, we find that 360 (90%) of these questions indeed suffer from the incomplete answer problem, which indicates that our automatic detection method is efficient. This randomly selected 400 questions along with their 2559 answers are then further manually summarized for evaluation of automatically generated answer summaries by our model in experiments.

<sup>4</sup><http://sandbox.yahoo.com/>

## 4.2 Evaluation Measures

When taking the summarization as a sequential bi-classification problem, we can make use of the usual precision, recall and F1 measures (Shen et al., 2007) for classification accuracy evaluation.

In our experiments, we also compare the precision, recall and F1 score in the ROUGE-1, ROUGE-2 and ROUGE-L measures (Lin, 2004) for answer summarization performance.

## 5 Experimental Results

### 5.1 Summarization Results

We adapt the Support Vector Machine (SVM) and Logistic Regression (LR) which have been reported to be effective for classification and the Linear CRF (LCRF) which is used to summarize ordinary text documents in (Shen et al., 2007) as baselines for comparison. To better illustrate the effectiveness of question segmentation based contextual factors and the group  $L_1$  regularization term, we carry the tests in the following sequence: (a) we use only the contextual factors  $cf_3$  and  $cf_4$  with default  $L_2$  regularization (gCRF); (b) we add the reply question based factors  $cf_1$  and  $cf_2$  to the model (gCRF-QS); and (c) we replace default  $L_2$  regularization with our proposed group  $L_1$  regularization term (gCRF-QS-11). For linear CRF system, we use all our textual and non-textual features as well as the local (exact previous and next) neighborhood contextual factors instead of the features of (Shen et al., 2007) for fairness. For the thresholds used in the contextual factors, we enforce  $\tau_{lq}$  to be equal to  $\tau_{ls}$  and  $\tau_{uq}$  equal to  $\tau_{us}$  for the purpose of simplifying the parameters setting ( $\tau_{lq} = \tau_{ls} = 0.4$ ,  $\tau_{uq} = \tau_{us} = 0.8$  in our experiments). We randomly divide the dataset into ten subsets (every subset with 40 questions and the associated answers), and conduct a ten-fold cross validation and for each round where the nine subsets are used to train the model and the remaining one for testing. The precision, recall and F1 measures of these models are presented in Table 2.

Table 2 shows that our general CRF model based on question segmentation with group  $L_1$  regularization out-performs the baselines significantly in all three measures (gCRF-QS-11 is 13.99% better than SVM in precision, 9.77% better in recall and 11.72% better in F1 score). We note that both SVM and LR,



Model	R1_P	R1_R	R1_F1	R2_P	R2_R	R2_F1	RL_P	RL_R	RL_F1
SVM	79.2%	52.5%	63.1%	71.9%	41.3%	52.4%	67.1%	36.7%	47.4%
LR	75.2%↓	57.4%↑	65.1%↑	66.1%↓	48.5%↑	56.0%↑	61.6%↓	43.2%↑	50.8%↑
LCRF	78.7%-	61.8%↑	69.3%-	71.4%-	54.1%↑	61.6%↑	67.1%-	49.6%↑	57.0%↑
<b>gCRF</b>	81.9%↑	65.2%↑	72.6%↑	76.8%↑	57.3%↑	65.7%↑	73.9%↑	53.5%↑	62.1%↑
<b>gCRF-QS</b>	81.4%-	<b>70.0%</b> ↑	75.3%↑	76.2%-	<b>62.4%</b> ↑	68.6%↑	73.3%-	<b>58.6%</b> ↑	65.1%↑
<b>gCRF-QS-II</b>	<b>86.6%</b> ↑	68.3%-	<b>76.4%</b> ↑	<b>82.6%</b> ↑	61.5%-	<b>70.5%</b> ↑	<b>80.4%</b> ↑	58.2%-	<b>67.5%</b> ↑

Table 3: The Precision, Recall and F1 of ROUGE-1, ROUGE-2, ROUGE-L in the baselines SVM,LR, LCRF and our general CRF based models (gCRF, gCRF-QS, gCRF-QS-II). The down-arrow means performance degradation with statistical significance.

Model	Precision	Recall	F1
SVM	65.93%	61.96%	63.88%
LR	66.92%-	61.31%-	63.99%-
LCRF	69.80%↑	63.91%-	66.73%↑
<b>gCRF</b>	73.77%↑	69.43%↑	71.53%↑
<b>gCRF-QS</b>	74.78%↑	<b>72.51%</b> ↑	73.63%↑
<b>gCRF-QS-II</b>	<b>79.92%</b> ↑	71.73%-	<b>75.60%</b> ↑

Table 2: The Precision, Recall and F1 measures of the baselines SVM,LR, LCRF and our general CRF based models (gCRF, gCRF-QS, gCRF-QS-II). The up-arrow denotes the performance improvement compared to the previous method (above) with statistical significance under p value of 0.05, the short line '-' denotes there is no difference in statistical significance.

which just utilize the independent sentence-level features, behave not vary well here, and there is no statistically significant performance difference between them. We also find that LCRF which utilizes the local context information between sentences perform better than the LR method in precision and F1 with statistical significance. While we consider the general local and non-local contextual factor  $cf_3$  and  $cf_4$  for novelty and non-redundancy constraints, the gCRF performs much better than LCRF in all three measures; and we obtain further performance improvement by adding the contextual factors based on QS, especially in the recall measurement. This is mainly because we have divided the question into several sub questions, and the system is able to select more novel sentences than just treating the original multi-sentence as a whole. In addition, when we replace the default  $L_2$  regularization by the group  $L_1$  regularization for more efficient feature weight learning, we obtain a much better performance in

precision while not sacrificing the recall measurement statistically.

We also compute the Precision, Recall and F1 in ROUGE-1, ROUGE-2 and ROUGE-L measurements, which are widely used to measure the quality of automatic text summarization. The experimental results are listed in Table 3. All results in the Table are the average of the ten-fold cross validation experiments on our dataset.

It is observed that our gCRF-QS-II model improves the performance in terms of precision, recall and F1 score on all three measurements of ROUGE-1, ROUGE-2 and ROUGE-L by a significant margin compared to other baselines due to the use of local and non-local contextual factors and factors based on QS with group  $L_1$  regularization. Since the ROUGE measures care more about the recall and precision of N-grams as well as common substrings to the reference summary rather than the whole sentence, they offer a better measurement in modeling the user’s information needs. Therefore, the improvements in these measures are more encouraging than those of the average classification accuracy for answer summarization.

From the viewpoint of ROUGE measures we observe that our question segmentation method can enhance the recall of the summaries significantly due to the more fine-grained modeling of sub questions. We also find that the precision of the group  $L_1$  regularization is much better than that of the default  $L_2$  regularization while not hurting the recall significantly. In general, the experimental results show that our proposed method is more effective than other baselines in answer summarization for addressing the incomplete answer problem in cQAs.

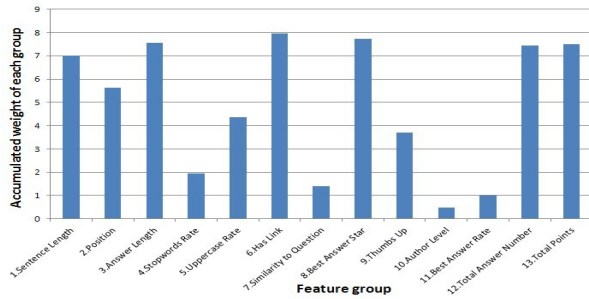


Figure 2: The accumulated weight of each site feature group in the group  $L_1$ -regularization to our Yahoo! Answer dataset. The horizontal axis corresponds to the name of each feature group.

## 5.2 Evaluation of Feature Learning

For group  $L_1$  regularization term, we set the  $\varepsilon = 10^{-4}$  in Equation 6. To see how much the different textual and non-textual features contribute to community answer summarization, the accumulated weight of each group of sentence-level features<sup>5</sup> is presented in Figure 2. It shows that the textual features such as 1 (*Sentence Length*), 2 (*Position*) 3 (*Answer Length*), 6 (*Has Link*) and non-textual features such as 8 (*Best Answer Star*), 12 (*Total Answer Number*) as well as 13 (*Total Points*) have larger weights, which play a significant role in the summarization task as we intuitively considered; features 4 (*Stopwords Rate*), 5 (*Uppercase Rate*) and 9 (*Thumbs Up*) have medium weights relatively; and the other features like 7 (*Similarity to Question*), 10 (*Author Level*) and 11 (*Best Answer Rate*) have the smallest accumulated weights. The main reasons that the feature 7 (*Similarity to Question*) has low contribution is that we have utilized the similarity to question in the contextual factors, and this similarity feature in the single site becomes redundant. Similarly, the features *Author Level* and *Best Answer Number* are likely to be redundant when other non-textual features (*Total Answer Number* and *Total Points*) are presented together. The experimental results demonstrate that with the use of group  $L_1$ -regularization we have learnt better combination of these features.

<sup>5</sup>Note that we have already evaluated the contribution of the contextual factors in Section 5.1.

## 5.3 An Example of Summarized Answer

To demonstrate the effectiveness of our proposed method, Table 4 shows the generated summary of the example question which is previously illustrated in Table 1 in the introduction section. The best answer available in the system and the summarized answer generated by our model are compared in Table 4. It is found that the summarized answer contains more valuable information about the original multi-sentence question, as it better answers the reason of teeth bleeding and offers some solution for it. Storing and indexing this summarized answer in question archives should provide a better choice for answer reuse in question retrieval of cQAs.

Question
Why do teeth bleed at night and how do you prevent/stop it? This morning I woke up with blood caked between my two front teeth.[...]
Best Answer - Chosen by Asker
Periodontal disease is a possibility, gingivitis, or some gum infection. Teeth don't bleed; gums bleed.
Summarized Answer Generated by Our Method
Periodontal disease is a possibility, gingivitis, or some gum infection. Teeth don't bleed; gums bleed. Gums that bleed could be a sign of a more serious issue like leukemia, an infection, gum disease, a blood disorder, or a vitamin deficiency. wash your mouth with warm water and salt, it will help to strengthen your gum and teeth, also salt avoid infection.

Table 4: Summarized answer by our general CRF based model for the question in Table 1.

## 6 Conclusions

We proposed a general CRF based community answer summarization method to deal with the incomplete answer problem for deep understanding of complex multi-sentence questions. Our main contributions are that we proposed a systematic way for modeling semantic contextual interactions between the answer sentences based on question segmentation and we explored both the textual and non-textual answer features learned via a group  $L_1$  regularization. We showed that our method is able to achieve significant improvements in performance of answer summarization compared to other baselines and previous methods on Yahoo! Answers dataset. We plan to extend our proposed model with more advanced feature learning as well as enriching our summarized answer with more available Web re-

sources.

## Acknowledgements

This work was supported by the NSFC under Grant No.61073002 and No.60773077.

## References

- L. A. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman. 2008. Knowledge sharing and yahoo answers: everyone knows something. *Proceedings of WWW 2008*.
- Shilin Ding, Gao Cong, Chin-Yew Lin and Xiaoyan Zhu. 2008. Rouge: Using Conditional Random Fields to Extract Contexts and Answers of Questions from Online Forums. *Proceedings of ACL-08: HLT*, pages 710–718.
- Michel Galley. 2006. A Skip-Chain Conditional Random Field for Ranking Meeting Utterances by Importance. *Proceedings of EMNLP 2006*.
- Z. Gyongyi, G. Koutrika, J. Pedersen, and H. Garcia-Molina. 2007. Questioning yahoo! answers. Technical report. *Stanford InfoLab*.
- F. Maxwell Harper, Daphne Raban, Sheizaf Rafaeli, and Joseph A. Konstan. 2008. Predictors of Answer Quality in Online Q&A Sites. *Proceedings of CHI 2008*.
- Liangjie Hong and Brian D. Davison. 2009. A Classification-based Approach to Question Answering in Discussion Boards. *Proceedings of the 32th ACM SIGIR Conference*, pages 171–178.
- Eduard Hovy, Chin Y. Lin, and Liang Zhou. 2005. A BE-based Multi-document Summarization with Sentence Compression. *Proceedings of Multilingual Summarization Evaluation (ACL 2005 workshop)*.
- Jiwoon Jeon, W. Bruce Croft, Joon Ho Lee and Soyeon Park. 2006. A Framework to Predict the Quality of Answers with NonTextual Features. *Proceedings of the 29th ACM SIGIR Conference*, pages 228–235.
- V. Jijkoun and M. de Rijke. 2005. Retrieving answers from frequently asked questions pages on the web. *In CIKM*.
- Daniel Jurafsky and James H. Martin. 2009. *Speech and Language Processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Published by Pearson Education.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of the 18th ICML*, pages 282–289.
- S. Lee, H. Lee, P. Abbeel, and A. Ng. 2006. Efficient L1 Regularized Logistic Regression. *In AAAI*.
- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Proceedings of ACL Workshop*, pages 74–81.
- Yandong Liu, Jiang Bian, and Eugene Agichtein. 2008. Predicting Information Seeker Satisfaction in Community Question Answering. *Proceedings of the 31st ACM SIGIR Conference*.
- Yuanjie Liu, Shasha Li, Yunbo Cao, Chin-Yew Lin, Dingyi Han, and Yong Yu. 2008. Understanding and summarizing answers in community-based question answering services. *Proceedings of the 22nd ICCL*, pages 497–504.
- S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu. 2007. Statistical machine translation for query expansion in answer retrieval. *Proceedings of the 45th Annual Meeting of ACL*.
- Mark Schmidt. 2010. Graphical Model Structure Learning with L1-Regularization. *Doctoral Thesis*.
- Chirag Shah and Jefferey Pomerantz. 2010. Evaluating and Predicting Answer Quality in Community QA. *Proceedings of the 33th ACM SIGIR Conference*.
- Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang and Zheng Chen. 2007. Document Summarization using Conditional Random Fields. *Proceedings of the 20th IJCAI*.
- Troy Simpson and Malcolm Crowe. 2005. WordNet.Net <http://opensource.ebswift.com/WordNet.Net>
- Mineki Takechi, Takenobu Tokunaga, and Yuji Matsumoto. 2007. Chunking-based Question Type Identification for Multi-Sentence Queries. *Proceedings of SIGIR 2007 Workshop*.
- Mattia Tomasoni and Minlie Huang. 2010. Metadata-Aware Measures for Answer Summarization in Community Question Answering. *Proceedings of the 48th Annual Meeting of ACL*, pages 760–769.
- Hongning Wang, Chi Wang, ChengXiang Zhai, Jiawei Han. 2011. Learning Online Discussion Structures by Conditional Random Fields. *Proceedings of the 34th ACM SIGIR Conference*.
- Kai Wang, Zhao-Yan Ming and Tat-Seng Chua. 2009. A Syntactic Tree Matching Approach to Finding Similar Questions in Community-based QA Services. *Proceedings of the 32th ACM SIGIR Conference*.
- Kai Wang, Zhao-Yan Ming, Xia Hu and Tat-Seng Chua. 2010. Segmentation of Multi-Sentence Questions: Towards Effective Question Retrieval in cQA Services. *Proceedings of the 33th ACM SIGIR Conference*, pages 387–394.
- X. Xue, J.Jeon, and W.B.Croft. 2008. Retrieval models for question and answers archives. *Proceedings of the 31th ACM SIGIR Conference*.
- Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhou Su, and Juanzi Li. 2011. Social Context Summarization. *Proceedings of the 34th ACM SIGIR Conference*.
- Liang Zhou, Chin Y. Lin, and Eduard Hovy. 2006. Summarizing answers for complicated questions. *Proceedings of the 5th International Conference on LREC, Genoa, Italy*.

# Error Mining on Dependency Trees

**Claire Gardent**

CNRS, LORIA, UMR 7503  
Vandoeuvre-lès-Nancy, F-54500, France  
claire.gardent@loria.fr

**Shashi Narayan**

Université de Lorraine, LORIA, UMR 7503  
Villers-lès-Nancy, F-54600, France  
shashi.narayan@loria.fr

## Abstract

In recent years, error mining approaches were developed to help identify the most likely sources of parsing failures in parsing systems using handcrafted grammars and lexicons. However the techniques they use to enumerate and count n-grams builds on the sequential nature of a text corpus and do not easily extend to structured data. In this paper, we propose an algorithm for mining trees and apply it to detect the most likely sources of generation failure. We show that this tree mining algorithm permits identifying not only errors in the generation system (grammar, lexicon) but also mismatches between the structures contained in the input and the input structures expected by our generator as well as a few idiosyncrasies/error in the input data.

## 1 Introduction

In recent years, error mining techniques have been developed to help identify the most likely sources of parsing failure (van Noord, 2004; Sagot and de la Clergerie, 2006; de Kok et al., 2009). First, the input data (text) is separated into two subcorpora, a corpus of sentences that could be parsed (PASS) and a corpus of sentences that failed to be parsed (FAIL). For each n-gram of words (and/or part of speech tag) occurring in the corpus to be parsed, a suspicion rate is then computed which, in essence, captures the likelihood that this n-gram causes parsing to fail.

These error mining techniques have been applied with good results on parsing output and shown to help improve the large scale symbolic grammars and

lexicons used by the parser. However the techniques they use (e.g., suffix arrays) to enumerate and count n-grams builds on the sequential nature of a text corpus and cannot easily extend to structured data.

There are some NLP applications though where the processed data is structured data such as trees or graphs and which would benefit from error mining. For instance, when generating sentences from dependency trees, as was proposed recently in the Generation Challenge Surface Realisation Task (SR Task, (Belz et al., 2011)), it would be useful to be able to apply error mining on the input trees to find the most likely causes of generation failure.

In this paper, we address this issue and propose an approach that supports error mining on trees. We adapt an existing algorithm for tree mining which we then use to mine the Generation Challenge dependency trees and identify the most likely causes of generation failure. We show in particular, that this tree mining algorithm permits identifying not only errors in the grammar and the lexicon used by generation but also a few idiosyncrasies/error in the input data as well as mismatches between the structures contained in the SR input and the input structures expected by our generator. The latter is an important point since, for symbolic approaches, a major hurdle to participation in the SR challenge is known to be precisely these mismatches i.e., the fact that the input provided by the SR task fails to match the input expected by the symbolic generation systems (Belz et al., 2011).

The paper is structured as follows. Section 2 presents the HybridTreeMiner algorithm, a complete and computationally efficient algorithm developed

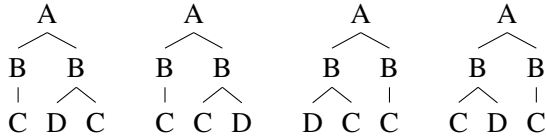


Figure 1: Four unordered labelled trees. The rightmost is in Breadth-First Canonical Form

by (Chi et al., 2004) for discovering frequently occurring subtrees in a database of labelled unordered trees. Section 3 shows how to adapt this algorithm to mine the SR dependency trees for subtrees with high suspicion rate. Section 4 presents an experiment we made using the resulting tree mining algorithm on SR dependency trees and summarises the results. Section 5 discusses related work. Section 6 concludes.

## 2 Mining Trees

Mining for frequent subtrees is an important problem that has many applications such as XML data mining, web usage analysis and RNA classification. The HybridTreeMiner (HTM) algorithm presented in (Chi et al., 2004) provides a complete and computationally efficient method for discovering frequently occurring subtrees in a database of labelled unordered trees and counting them. We now sketch the intuition underlying this algorithm<sup>1</sup>. In the next section, we will show how to modify this algorithm to mine for errors in dependency trees.

Given a set of trees  $T$ , the HybridTreeMiner algorithm proceeds in two steps. First, the unordered labelled trees contained in  $T$  are converted to a canonical form called BFCF (Breadth-First Canonical Form). In that way, distinct instantiations of the same unordered trees have a unique representation. Second, the subtrees of the BFCF trees are enumerated in increasing size order using two tree operations called join and extension and their support (the number of trees in the database that contains each subtree) is recorded. In effect, the algorithm builds an enumeration tree whose nodes are the possible subtrees of  $T$  and such that, at depth  $d$  of this enumeration tree, all possible frequent subtrees consisting of  $d$  nodes are listed.

<sup>1</sup>For a more complete definition see (Chi et al., 2004).

The BFCF canonical form of an unordered tree is an ordered tree  $t$  such that  $t$  has the smallest breath-first canonical string (BFCS) encoding according to lexicographic order. The BFCS encoding of a tree is obtained by breadth-first traversal of the tree, recording the string labelling each node, “\$” to separate siblings with distinct parents and “#” to represent the end of the tree<sup>2</sup>. For instance, the BFCS encodings of the four trees shown in Figure 1 are ‘A\$BB\$C\$DC#’, ‘A\$BB\$C\$CD#’, ‘A\$BB\$DC\$C#’ and ‘A\$BB\$CD\$C#’ respectively. Hence, the rightmost tree is the BFCF of all four trees.

The join and extension operations used to iteratively enumerate subtrees are depicted in Figure 2 and can be defined as follows.

- A leg is a leaf of maximal depth.
- Extension: Given a tree  $t$  of height  $h_t$  and a node  $n$ , extending  $t$  with  $n$  yields a tree  $t'$  (a child of  $t$  in the enumeration tree) with height  $h_{t'}$  such that  $n$  is a child of one of  $t$ 's legs and  $h_{t'}$  is  $h_t + 1$ .
- Join: Given two trees  $t_1$  and  $t_2$  of same height  $h$  differing only in their rightmost leg and such that  $t_1$  sorts lower than  $t_2$ , joining  $t_1$  and  $t_2$  yields a tree  $t'$  (a child of  $t_1$  in the enumeration tree) of same height  $h$  by adding the rightmost leg of  $t_2$  to  $t_1$  at level  $h - 1$ .

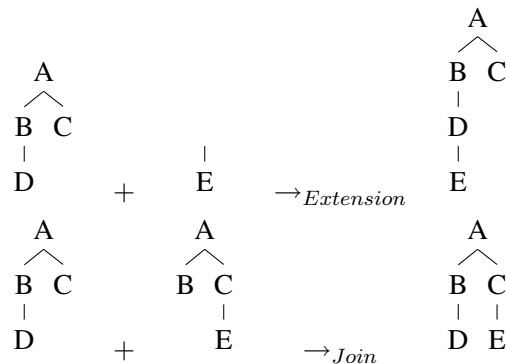


Figure 2: Join and Extension Operations

To support counting, the algorithm additionally records for each subtree a list (called *occurrence list*)

<sup>2</sup>Assuming “#” sorts greater than “\$” and both sort greater than any other alphabets in node labels.

of all trees in which this subtree occurs and of its position in the tree (represented by the list of tree nodes mapped onto by the subtree). Thus for a given subtree  $t$ , the *support* of  $t$  is the number of elements in that list. Occurrence lists are also used to check that trees that are combined occur in the data. For the join operation, the subtrees being combined must occur in the same tree at the same position (the intersection of their occurrence lists must be non empty and the tree nodes must match except the last node). For the extension operation, the extension of a tree  $t$  is licensed for any given occurrence in the occurrence list only if the planned extension maps onto the tree identified by the occurrence.

### 3 Mining Dependency Trees

We develop an algorithm (called ErrorTreeMiner, ETM) which adapts the HybridTreeMiner algorithm to mine sources of generation errors in the Generation Challenge SR shallow input data. The main modification is that instead of simply counting trees, we want to compute their suspicion rate. Following (de Kok et al., 2009), we take the suspicion rate of a given subtree  $t$  to be the proportion of cases where  $t$  occurs in an input tree for which generation fails:

$$Sus(t) = \frac{count(t|FAIL)}{count(t)}$$

where  $count(t)$  is the number of occurrences of  $t$  in all input trees and  $count(t|FAIL)$  is the number of occurrences of  $t$  in input trees for which no output was produced.

Since we work with subtrees of arbitrary length, we also need to check whether constructing a longer subtree is useful that is, whether its suspicion rate is equal or higher than the suspicion rate of any of the subtrees it contains. In that way, we avoid computing all subtrees (thus saving time and space). As noted in (de Kok et al., 2009), this also permits bypassing *suspicion sharing* that is the fact that, if  $n_2$  is the cause of a generation failure, and if  $n_2$  is contained in larger trees  $n_3$  and  $n_4$ , then all three trees will have high suspicion rate making it difficult to identify the actual source of failure namely  $n_2$ . Because we use a milder condition however (we accept bigger trees whose suspicion rate is *equal* to the suspicion rate of any of their subtrees), some amount of

---

#### Algorithm 1 ErrorTreeMiner( $D, minsup$ )

---

Note:  $D$  consists of  $D_{fail}$  and  $D_{pass}$   
 $F_1 \leftarrow \{\text{Frequent 1-trees}\}$   
 $F_2 \leftarrow \emptyset$   
**for**  $i \leftarrow 1, \dots, |F_1|$  **do**  
  **for**  $j \leftarrow 1, \dots, |F_1|$  **do**  
     $q \leftarrow f_i$  plus *leg* $f_j$   
    **if** Noord-Validation( $q, minsup$ ) **then**  
       $F_2 \leftarrow F_2 \cup q$   
    **end if**  
  **end for**  
**end for**  
 $F \leftarrow F_1 \cup F_2$   
PUSH:  $sort(F_2) \rightarrow L_{Queue}$   
Enum-Grow( $L_{Queue}, F, minsup$ )  
**return**  $F$

---



---

#### Algorithm 2 Enum-Grow( $L_{Queue}, F, minsup$ )

---

**while**  $L_{Queue} \neq empty$  **do**  
  POP:  $pop(L_{Queue}) \rightarrow C$   
  **for**  $i \leftarrow 1, \dots, |C|$  **do**  
    ◊The join operation  
     $J \leftarrow \emptyset$   
    **for**  $j \leftarrow i, \dots, |C|$  **do**  
       $p \leftarrow join(c_i, c_j)$   
      **if** Noord-Validation( $p, minsup$ ) **then**  
         $J \leftarrow J \cup p$   
      **end if**  
    **end for**  
     $F \leftarrow F \cup J$   
  PUSH:  $sort(J) \rightarrow L_{Queue}$   
  ◊The extension operation  
   $E \leftarrow \emptyset$   
  **for** possible leg  $l_m$  of  $c_i$  **do**  
    **for** possible new leg  $l_n \in F_1$  **do**  
       $q \leftarrow extend\ c_i\ with\ l_n\ at\ position\ l_m$   
      **if** Noord-Validation( $q, minsup$ ) **then**  
         $E \leftarrow E \cup q$   
      **end if**  
    **end for**  
  **end for**  
   $F \leftarrow F \cup E$   
  PUSH:  $sort(E) \rightarrow L_{Queue}$   
**end for**  
**end while**

---

---

**Algorithm 3** Noord-Validation( $t_n, minsup$ )

---

Note:  $t_n$ , tree with  $n$  nodes

```
if  $Sup(t_n) \geq minsup$  then  
  if  $Sus(t_n) \geq Sus(t_{n-1}), \forall t_{n-1}$  in  $t_n$  then  
    return true  
  end if  
end if  
return false
```

---

suspicion sharing remains. As we shall see in Section 4.3.2, relaxing this check though allows us to extract frequent larger tree patterns and thereby get a more precise picture of the context in which highly suspicious items occur.

Finally, we only keep subtrees whose support is above a given threshold where the support  $Sup(t)$  of a tree  $t$  is defined as the ratio between the number of times it occurs in an input for which generation fails and the total number of generation failures:

$$Sup(t) = \frac{count(t|FAIL)}{count(FAIL)}$$

The modified algorithm we use for error mining is given in Algorithm 1, 2 and 3. It can be summarised as follows.

First, dependency trees are converted to Breadth-First Canonical Form whereby lexicographic order can apply to the word forms labelling tree nodes, to their part of speech, to their dependency relation or to any combination thereof<sup>3</sup>.

Next, the algorithm iteratively enumerates the subtrees occurring in the input data in increasing size order and associating each subtree  $t$  with two occurrence lists namely, the list of input trees in which  $t$  occurs and for which generation was successful ( $PASS(t)$ ); and the list of input trees in which  $t$  occurs and for which generation failed ( $FAIL(t)$ ).

This process is initiated by building trees of size one (i.e., one-node tree) and extending them to trees of size two. It is then continued by extending the trees using the join and extension operations. As explained in Section 2 above, join and extension only apply provided the resulting trees occur in the data (this is checked by looking up occurrence lists).

---

<sup>3</sup>For convenience, the dependency relation labelling the edges of dependency trees is brought down to the daughter node of the edge.

Each time an  $n$ -node tree  $t_n$ , is built, it is checked that (i) its support is above the set threshold and (ii) its suspicion rate is higher than or equal to the suspicion rate of all  $(n - 1)$ -node subtrees of  $t_n$ .

In sum, the ETM algorithm differs from the HTM algorithm in two main ways. First, while HTM explores the enumeration tree depth-first, ETM proceeds breadth-first to ensure that the suspicion rate of  $(n-1)$ -node trees is always available when checking whether an  $n$ -node tree should be introduced. Second, while the HTM algorithm uses support to prune the search space (only trees with a minimum support bigger than the set threshold are stored), the ETM algorithm drastically prunes the search space by additionally checking that the suspicion rate of all subtrees contained in a new tree  $t$  is smaller or equal to the suspicion rate of  $t$ . As a result, while ETM loses the space advantage of HTM by a small margin<sup>4</sup>, it benefits from a much stronger pruning of the search space than HTM through suspicion rate checking. In practice, the ETM algorithm allows us to process e.g., all NP chunks of size 4 and 6 present in the SR data (roughly 60 000 trees) in roughly 20 minutes on a PC.

## 4 Experiment and Results

Using the input data provided by the Generation Challenge SR Task, we applied the error mining algorithm described in the preceding Section to debug and extend a symbolic surface realiser developed for this task.

### 4.1 Input Data and Surface Realisation System

The shallow input data provided by the SR Task was obtained from the Penn Treebank using the LTH Constituent-to-Dependency Conversion Tool for Penn-style Treebanks (Pennconverter, (Johansson and Nugues, 2007)). It consists of a set of unordered labelled syntactic dependency trees whose nodes are labelled with word forms, part of speech categories, partial morphosyntactic information such as tense and number and, in some cases, a sense tag identifier. The edges are labelled with the syntactic labels provided by the Pennconverter. All words (including punctuation) of the original sen-

---

<sup>4</sup>ETM needs to store all  $(n-1)$ -node trees in queues before producing  $n$ -node trees.

tence are represented by a node in the tree and the alignment between nodes and word forms was provided by the organisers.

The surface realiser used is a system based on a Feature-Based Lexicalised Tree Adjoining Grammar (FB-LTAG) for English extended with a unification based compositional semantics. Both the grammars and the lexicon were developed in view of the Generation Challenge and the data provided by this challenge was used as a means to debug and extend the system. Unknown words are assigned a default TAG family/tree based on the part of speech they are associated with in the SR data. The surface realisation algorithm extends the algorithm proposed in (Gardent and Perez-Beltrachini, 2010) and adapts it to work on the SR dependency input rather than on flat semantic representations.

## 4.2 Experimental Setup

To facilitate interpretation, we first chunked the input data in NPs, PPs and Clauses and performed error mining on the resulting sets of data. The chunking was performed by retrieving from the Penn Treebank (PTB), for each phrase type, the yields of the constituents of that type and by using the alignment between words and dependency tree nodes provided by the organisers of the SR Task. For instance, given the sentence “*The most troublesome report may be the August merchandise trade deficit due out tomorrow*”, the NPs “*The most troublesome report*” and “*the August merchandise trade deficit due out tomorrow*” will be extracted from the PTB and the corresponding dependency structures from the SR Task data.

Using this chunked data, we then ran the generator on the corresponding SR Task dependency trees and stored separately, the input dependency trees for which generation succeeded and the input dependency trees for which generation failed. Using information provided by the generator, we then removed from the failed data, those cases where generation failed either because a word was missing in the lexicon or because a TAG tree/family was missing in the grammar but required by the lexicon and the input data. These cases can easily be detected using the generation system and thus do not need to be handled by error mining.

Finally, we performed error mining on the data

using different minimal support thresholds, different display modes (sorted first by size and second by suspicion rate vs sorted by suspicion rate) and different labels (part of speech, words and part of speech, dependency, dependency and part of speech).

## 4.3 Results

One feature of our approach is that it permits mining the data for tree patterns of arbitrary size using different types of labelling information (POS tags, dependencies, word forms and any combination thereof). In what follows, we focus on the NP chunk data and illustrate by means of examples how these features can be exploited to extract complementary debugging information from the data.

### 4.3.1 Mining on single labels (word form, POS tag or dependency)

Mining on a single label permits (i) assessing the relative impact of each category in a given label category and (ii) identifying different sources of errors depending on the type of label considered (POS tag, dependency or word form).

**Mining on POS tags** Table 1 illustrates how mining on a single label (in this case, POS tags) gives a good overview of how the different categories in that label type impact generation: two POS tags (POS and CC) have a suspicion rate of 0.99 indicating that these categories always lead generation to fail. Other POS tag with much lower suspicion rate indicate that there are unresolved issues with, in decreasing order of suspicion rate, cardinal numbers (CD), proper names (NNP), nouns (NN), prepositions (IN) and determiners (DT).

The highest ranking category (POS<sup>5</sup>) points to a mismatch between the representation of genitive NPs (e.g., *John’s father*) in the SR Task data and in the grammar. While our generator expects the representation of ‘John’s father’ to be FATHER(“S”(JOHN)), the structure provided by the SR Task is FATHER(JOHN(“S”)). Hence whenever a possessive appears in the input data, generation fails. This is in line with (Rajkumar et al., 2011)’s finding that the logical forms expected by their system for possessives differed from the shared task inputs.

<sup>5</sup>In the Penn Treebank, the POS tag is the category assigned to possessive ‘s.



POS	Sus	Sup	Fail	Pass
POS	0.99	0.38	3237	1
CC	0.99	0.21	1774	9
CD	0.39	0.16	1419	2148
NNP	0.35	0.32	2749	5014
NN	0.30	0.81	6798	15663
IN	0.30	0.16	1355	3128
DT	0.09	0.12	1079	10254

Table 1: Error Mining on POS tags with frequency cutoff 0.1 and displaying only trees of size 1 sorted by decreasing suspicion rate (Sus)

The second highest ranked category is CC for coordinations. In this case, error mining unveils a bug in the grammar trees associated with conjunction which made all sentences containing a conjunction fail. Because the grammar is compiled out of a strongly factorised description, errors in this description can propagate to a large number of trees in the grammar. It turned out that an error occurred in a class inherited by all conjunction trees thereby blocking the generation of any sentence requiring the use of a conjunction.

Next but with a much lower suspicion rate come cardinal numbers (CD), proper names (NNP), nouns (NN), prepositions (IN) and determiners (DT). We will see below how the richer information provided by mining for larger tree patterns with mixed labelling information permits identifying the contexts in which these POS tags lead to generation failure.

**Mining on Word Forms** Because we remove from the failure set all cases of errors due to a missing word form in the lexicon, a high suspicion rate for a word form usually indicates a missing or incorrect lexical entry: the word is present in the lexicon but associated with either the wrong POS tag and/or the wrong TAG tree/family. To capture such cases, we therefore mine not on word forms alone but on pairs of word forms and POS tag. In this way, we found for instance, that cardinal numbers induced many generation failures whenever they were categorised as determiners but not as nouns in our lexicon. As we will see below, larger tree patterns help identify the specific contexts inducing such failures.

One interesting case stood out which pointed to idiosyncrasies in the input data: The word form \$

(Sus=1) was assigned the POS tag \$ in the input data, a POS tag which is unknown to our system and not documented in the SR Task guidelines. The SR guidelines specify that the Penn Treebank tagset is used modulo the modifications which are explicitly listed. However for the \$ symbol, the Penn treebank used SYM as a POS tag and the SR Task \$, but the modification is not listed. Similarly, while in the Penn treebank, punctuations are assigned the SYM POS tag, in the SR data “,” is used for the comma, “(“ for an opening bracket and so on.

**Mining on Dependencies** When mining on dependencies, suspects can point to syntactic constructions (rather than words or word categories) that are not easily spotted when mining on words or parts of speech. Thus, while problems with coordination could easily be spotted through a high suspicion rate for the CC POS tag, some constructions are linked neither to a specific POS tag nor to a specific word. This is the case, for instance, for apposition which a suspicion rate of 0.19 (286F/1148P) identified as problematic. Similarly, a high suspicion rate (0.54, 183F/155P) on the TMP dependency indicates that temporal modifiers are not correctly handled either because of missing or erroneous information in the grammar or because of a mismatch between the input data and the format expected by the surface realiser.

Interestingly, the underspecified dependency relation DEP which is typically used in cases for which no obvious syntactic dependency comes to mind shows a suspicion rate of 0.61 (595F/371P).

#### 4.3.2 Mining on trees of arbitrary size and complex labelling patterns

While error mining with tree patterns of size one permits ranking and qualifying the various sources of errors, larger patterns often provide more detailed contextual information about these errors. For instance, Table 1 shows that the CD POS tag has a suspicion rate of 0.39 (1419F/2148P). The larger tree patterns identified below permits a more specific characterization of the context in which this POS tag co-occurs with generation failure:

TP1	CD(IN,RBR)	<i>more than 10</i>
TP2	IN(CD)	<i>of 1991</i>
TP3	NNP(CD)	<i>November 1</i>
TP4	CD(NNP(CD))	<i>Nov. 1, 1997</i>

Two patterns clearly emerge: a pattern where cardinal numbers are parts of a date (tree patterns TP2-TP4) and a more specific pattern (TP1) involving the comparative construction (e.g., *more than 10*). All these patterns in fact point to a missing category for cardinals in the lexicon: they are only associated with determiner TAG trees, not nouns, and therefore fail to combine with prepositions (e.g., *of 1991, than 10*) and with proper names (e.g., *November 1*).

For proper names (NNP), dates also show up because months are tagged as proper names (TP3,TP4) as well as addresses TP5:

TP5 NNP(“,”“,”) *Brooklyn, n.y.*,

For prepositions (IN), we find, in addition to the TP1-TP2, the following two main patterns:

TP6 DT(IN) *those with, some of*

TP7 RB(IN) *just under, little more*

Pattern TP6 points to a missing entry for words such as *those* and *some* which are categorised in the lexicon as determiners but not as nouns. TP7 points to a mismatch between the SR data and the format expected by the generator: while the latter expects the structure IN(RB), the input format provided by the SR Task is RB(IN).

#### 4.4 Improving Generation Using the Results of Error Mining

Table 2 shows how implementing some of the corrections suggested by error mining impacts the number of NP chunks (size 4) that can be generated. In this experiment, the total number of input (NP) dependency trees is 24995. Before error mining, generation failed on 33% of these input. Correcting the erroneous class inherited by all conjunction trees mentioned in Section 4.3.1 brings generation failure down to 26%. Converting the input data to the correct input format to resolve the mismatch induced by possessive 's (cf. Section 4.3.1) reduce generation failure to 21%<sup>6</sup> and combining both corrections results in a failure rate of 13%. In other words, error mining permits quickly identifying two issues which, once corrected, reduces generation failure by 20 points.

When mining on clause size chunks, other mismatches were identified such as in particular, mismatches introduced by subjects and auxiliaries:

<sup>6</sup>For NP of size 4, 3264 structures with possessive 's were rewritten.

NP 4	Before	After
SR Data	8361	6511
Rewritten SR Data	5255	3401

Table 2: Diminishing the number of errors using information from error mining. The table compares the number of failures on NP chunks of size 4 before (first row) and after (second row) rewriting the SR data to the format expected by our generator and before (second column) and after (third column) correcting the grammar and lexicon errors discussed in Section 4.3.1

while our generator expects both the subject and the auxiliary to be children of the verb, the SR data represent the subject and the verb as children of the auxiliary.

## 5 Related Work

We now relate our proposal (i) to previous proposals on error mining and (ii) to the use of error mining in natural language generation.

**Previous work on error mining.** (van Noord, 2004) initiated error mining on parsing results with a very simple approach computing the parsability rate of each n-gram in a very large corpus. The parsability rate of an n-gram  $w_1 \dots w_n$  is the ratio  $R(w_1 \dots w_n) = \frac{C(w_1 \dots w_n | OK)}{C(w_1 \dots w_n)}$  with  $C(w_1 \dots w_n)$  the number of sentences in which the n-gram  $w_1 \dots w_n$  occurs and  $C(w_1 \dots w_n | OK)$  the number of sentences containing  $w_1 \dots w_n$  which could be parsed. The corpus is stored in a suffix array and the sorted suffixes are used to compute the frequency of each n-grams in the total corpus and in the corpus of parsed sentences. The approach was later extended and refined in (Sagot and de la Clergerie, 2006) and (de Kok et al., 2009) whereby (Sagot and de la Clergerie, 2006) defines a suspicion rate for n-grams which takes into account the number of occurrences of a given word form and iteratively defines the suspicion rate of each word form in a sentence based on the suspicion rate of this word form in the corpus; (de Kok et al., 2009) combined the iterative error mining proposed by (Sagot and de la Clergerie, 2006) with expansion of forms to n-grams of words and POS tags of arbitrary length.

Our approach differs from these previous ap-

proaches in several ways. First, error mining is performed on trees. Second, it can be parameterised to use any combination of POS tag, dependency and/or word form information. Third, it is applied to generation input rather than parsing output. Typically, the input to surface realisation is a structured representation (i.e., a flat semantic representation, a first order logic formula or a dependency tree) rather than a string. Mining these structured representations thus permits identifying causes of undergeneration in surface realisation systems.

**Error Mining for Generation** Not much work has been done on mining the results of surface realisers. Nonetheless, (Gardent and Kow, 2007) describes an error mining approach which works on the output of surface realisation (the generated sentences), manually separates correct from incorrect output and looks for derivation items which systematically occur in incorrect output but not in correct ones. In contrast, our approach works on the input to surface realisation, automatically separates correct from incorrect items using surface realisation and targets the most likely sources of errors rather than the absolute ones.

More generally, our approach is the first to our knowledge, which mines a surface realiser for undergeneration. Indeed, apart from (Gardent and Kow, 2007), most previous work on surface realisation evaluation has focused on evaluating the performance and the coverage of surface realisers. Approaches based on reversible grammars (Carroll et al., 1999) have used the semantic formulae output by parsing to evaluate the coverage and performance of their realiser; similarly, (Gardent et al., 2010) developed a tool called GenSem which traverses the grammar to produce flat semantic representations and thereby provide a benchmark for performance and coverage evaluation. In both cases however, because it is produced using the grammar exploited by the surface realiser, the input produced can only be used to test for overgeneration (and performance). (Callaway, 2003) avoids this shortcoming by converting the Penn Treebank to the format expected by his realiser. However, this involves manually identifying the mismatches between two formats much like symbolic systems did in the Generation Challenge SR Task. The error mining approach we pro-

pose helps identifying such mismatches automatically.

## 6 Conclusion

Previous work on error mining has focused on applications (parsing) where the input data is sequential working mainly on words and part of speech tags. In this paper, we proposed a novel approach to error mining which permits mining trees. We applied it to the input data provided by the Generation Challenge SR Task. And we showed that this supports the identification of gaps and errors in the grammar and in the lexicon; and of mismatches between the input data format and the format expected by our realiser.

We applied our error mining approach to the input of a surface realiser to identify the most likely sources of *undergeneration*. We plan to also explore how it can be used to detect the most likely sources of *overgeneration* based on the output of this surface realiser on the SR Task data. Using the Penn Treebank sentences associated with each SR Task dependency tree, we will create the two tree sets necessary to support error mining by dividing the set of trees output by the surface realiser into a set of trees (FAIL) associated with overgeneration (the generated sentences do not match the original sentences) and a set of trees (SUCCESS) associated with success (the generated sentence matches the original sentences). Exactly which tree should populate the SUCCESS and FAIL set is an open question. The various evaluation metrics used by the SR Task (BLEU, NIST, METEOR and TER) could be used to determine a threshold under which an output is considered incorrect (and thus classified as FAIL). Alternatively, a strict matching might be required. Similarly, since the surface realiser is non deterministic, the number of output trees to be kept will need to be experimented with.

## Acknowledgments

We would like to thank Clément Jacq for useful discussions on the hybrid tree miner algorithm. The research presented in this paper was partially supported by the European Fund for Regional Development within the framework of the INTERREG IV A Allegro Project.

## References

- Anja Belz, Michael White, Dominic Espinosa, Eric Kow, Deirdre Hogan, and Amanda Stent. 2011. The first surface realisation shared task: Overview and evaluation results. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, Nancy, France.
- Charles B. Callaway. 2003. Evaluating coverage for large symbolic NLG grammars. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 811–817, Acapulco, Mexico.
- John Carroll, Ann Copestake, Dan Flickinger, and Viktor Paznański. 1999. An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of the 7th European Workshop on Natural Language Generation*, pages 86–95, Toulouse, France.
- Yun Chi, Yirong Yang, and Richard R. Muntz. 2004. Hybridtreeminer: An efficient algorithm for mining frequent rooted trees and free trees using canonical form. In *Proceedings of the 16th International Conference on and Statistical Database Management (SS-DBM)*, pages 11–20, Santorini Island, Greece. IEEE Computer Society.
- Daniël de Kok, Jianqiang Ma, and Gertjan van Noord. 2009. A generalized method for iterative error mining in parsing results. In *Proceedings of the 2009 Workshop on Grammar Engineering Across Frameworks (GEAF 2009)*, pages 71–79, Suntec, Singapore. Association for Computational Linguistics.
- Claire Gardent and Eric Kow. 2007. Spotting overgeneration suspect. In *Proceedings of the 11th European Workshop on Natural Language Generation (ENLG)*, pages 41–48, Schloss Dagstuhl, Germany.
- Claire Gardent and Laura Perez-Beltrachini. 2010. Rtg based surface realisation for tag. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, pages 367–375, Beijing, China.
- Claire Gardent, Benjamin Gottesman, and Laura Perez-Beltrachini. 2010. Comparing the performance of two TAG-based Surface Realisers using controlled Grammar Traversal. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING - Poster session)*, pages 338–346, Beijing, China.
- Richert Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for english. In *Proceedings of the 16th Nordic Conference of Computational Linguistics (NODALIDA)*, pages 105–112, Tartu, Estonia.
- Rajakrishnan Rajkumar, Dominic Espinosa, and Michael White. 2011. The osu system for surface realization at generation challenges 2011. In *Proceedings of the 13th European Workshop on Natural Language Generation (ENLG)*, pages 236–238, Nancy, France.
- Benoît Sagot and Éric de la Clergerie. 2006. Error mining in parsing results. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 329–336, Sydney, Australia.
- Gertjan van Noord. 2004. Error mining for wide-coverage grammar engineering. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, pages 446–453, Barcelona, Spain.

# Computational Approaches to Sentence Completion

**Geoffrey Zweig, John C. Platt**  
**Christopher Meek**  
**Christopher J.C. Burges**  
Microsoft Research  
Redmond, WA 98052

**Ainur Yessenalina**  
Cornell University  
Computer Science Dept.  
Ithaca, NY 14853

**Qiang Liu**  
Univ. of California, Irvine  
Info. & Comp. Sci.  
Irvine, California 92697

## Abstract

This paper studies the problem of sentence-level semantic coherence by answering SAT-style sentence completion questions. These questions test the ability of algorithms to distinguish sense from nonsense based on a variety of sentence-level phenomena. We tackle the problem with two approaches: methods that use local lexical information, such as the n-grams of a classical language model; and methods that evaluate global coherence, such as latent semantic analysis. We evaluate these methods on a suite of practice SAT questions, and on a recently released sentence completion task based on data taken from five Conan Doyle novels. We find that by fusing local and global information, we can exceed 50% on this task (chance baseline is 20%), and we suggest some avenues for further research.

## 1 Introduction

In recent years, standardized examinations have proved a fertile source of evaluation data for language processing tasks. They are valuable for many reasons: they represent facets of language understanding recognized as important by educational experts; they are organized in various formats designed to evaluate specific capabilities; they are yardsticks by which society measures educational progress; and they affect a large number of people.

Previous researchers have taken advantage of this material to test both narrow and general language processing capabilities. Among the narrower tasks, the identification of synonyms and antonyms has

been studied by (Landauer and Dumais, 1997; Mohammed et al., 2008; Mohammed et al., 2011; Turney et al., 2003; Turney, 2008), who used questions from the Test of English as a Foreign Language (TOEFL), Graduate Record Exams (GRE) and English as a Second Language (ESL) exams. Tasks requiring broader competencies include logic puzzles and reading comprehension. Logic puzzles drawn from the Law School Administration Test (LSAT) and the GRE were studied in (Lev et al., 2004), which combined an extensive array of techniques to solve the problems. The DeepRead system (Hirschman et al., 1999) initiated a long line of research into reading comprehension based on test prep material (Charniak et al., 2000; Riloff and Theilen, 2000; Wang et al., 2000; Ng et al., 2000).

In this paper, we study a new class of problems intermediate in difficulty between the extremes of synonym detection and general question answering - the sentence completion questions found on the Scholastic Aptitude Test (SAT). These questions present a sentence with one or two blanks that need to be filled in. Five possible words (or short phrases) are given as options for each blank. All possible answers except one result in a nonsense sentence. Two examples are shown in Figure 1.

The questions are highly constrained in the sense that all the information necessary is present in the sentence itself without any other context. Nevertheless, they vary widely in difficulty. The first of these examples is relatively simple: the second half of the sentence is a clear description of the type of behavior characterized by the desired adjective. The second example is more sophisticated; one must infer from

1. One of the characters in Milton Murayama's novel is considered \_\_\_\_\_ because he deliberately defies an oppressive hierarchical society.  
(A) rebellious (B) impulsive (C) artistic (D) industrious (E) tyrannical
  
2. Whether substances are medicines or poisons often depends on dosage, for substances that are \_\_\_\_\_ in small doses can be \_\_\_\_\_ in large.  
(A) useless .. effective  
(B) mild .. benign  
(C) curative .. toxic  
(D) harmful .. fatal  
(E) beneficial .. miraculous

Figure 1: Sample sentence completion questions (Educational-Testing-Service, 2011).

the contrast between medicine and poison that the correct answer involves a contrast, either *useless vs. effective* or *curative vs. toxic*. Moreover, the first, incorrect, possibility is perfectly acceptable in the context of the second clause alone; only irrelevance to the contrast between medicine and poison eliminates it. In general, the questions require a combination of semantic and world knowledge as well as occasional logical reasoning. We study the sentence completion task because we believe it is complex enough to pose a significant challenge, yet structured enough that progress may be possible.

As a first step, we have approached the problem from two points-of-view: first by exploiting local sentence structure, and secondly by measuring a novel form of global sentence coherence based on latent semantic analysis. To investigate the usefulness of local information, we evaluated n-gram language model scores, from both a conventional model with Good-Turing smoothing, and with a recently proposed maximum-entropy class-based n-gram model (Chen, 2009a; Chen, 2009b). Also in the language modeling vein, but with potentially global context, we evaluate the use of a recurrent neural network language model. In all the language modeling approaches, a model is used to compute a sentence probability with each of the potential completions. To measure global coherence, we propose

a novel method based on latent semantic analysis (LSA). We find that the LSA based method performs best, and that both local and global information can be combined to exceed 50% accuracy. We report results on a set of questions taken from a collection of SAT practice exams (Princeton-Review, 2010), and further validate the methods with the recently proposed MSR Sentence Completion Challenge set (Zweig and Burges, 2011).

Our paper thus makes the following contributions: First, we present the first published results on the SAT sentence completion task. Secondly, we evaluate the effectiveness of both local n-gram information, and global coherence in the form of a novel LSA-based metric. Finally, we illustrate that the local and global information can be effectively fused.

The remainder of this paper is organized as follows. In Section 2 we discuss related work. Section 3 describes the language modeling methods we have evaluated. Section 4 outlines the LSA-based methods. Section 5 presents our experimental results. We conclude with a discussion in Section 6.

## 2 Related Work

The past work which is most similar to ours is derived from the lexical substitution track of SemEval-2007 (McCarthy and Navigli, 2007). In this task, the challenge is to find a replacement for a word or phrase removed from a sentence. In contrast to our SAT-inspired task, the original answer is indicated. For example, one might be asked to find alternates for *match* in "After the *match*, replace any remaining fluid deficit to prevent problems of chronic dehydration throughout the tournament." Two consistently high-performing systems for this task are the KU (Yuret, 2007) and UNT (Hassan et al., 2007) systems. These operate in two phases: first they find a set of potential replacement words, and then they rank them. The KU system uses just an N-gram language model to do this ranking. The UNT system uses a large variety of information sources, and a language model score receives the highest weight. N-gram statistics were also very effective in (Giuliano et al., 2007). That paper also explores the use of Latent Semantic Analysis to measure the degree of similarity between a potential replacement and its context, but the results are poorer than others. Since the original word provides a strong hint as to the pos-

sible meanings of the replacements, we hypothesize that N-gram statistics are largely able to resolve the remaining ambiguities. The SAT sentence completion sentences do not have this property and thus are more challenging.

Related to, but predating the Semeval lexical substitution task are the ESL synonym questions proposed by Turney (2001), and subsequently considered by numerous research groups including Terra and Clarke (2003) and Pado and Lapata (2007). These questions are similar to the SemEval task, but in addition to the original word and the sentence context, the list of options is provided. Jarmasz and Szpakowicz (2003) used a sophisticated thesaurus-based method and achieved state-of-the-art performance, which is 82%.

Other work on standardized tests includes the synonym and antonym tasks mentioned in Section 1, and more recent work on a SAT analogy task introduced by (Turney et al., 2003) and extensively used by other researchers (Veale, 2004; Turney and Littman, 2005; D. et al., 2009).

### 3 Sentence Completion via Language Modeling

Perhaps the most straightforward approach to solving the sentence completion task is to form the complete sentence with each option in turn, and to evaluate its likelihood under a language model. As discussed in Section 2, this was found to be very effective in the ranking phase of several SemEval systems. In this section, we describe the suite of state-of-the-art language modeling techniques for which we will present results. We begin with n-gram models; first a classical n-gram backoff model (Chen and Goodman, 1999), and then a recently proposed class-based maximum-entropy n-gram model (Chen, 2009a; Chen, 2009b). N-gram models have the obvious disadvantage of using a very limited context in predicting word probabilities. Therefore we evaluate the recurrent neural net model of (Mikolov et al., 2010; Mikolov et al., 2011b). This model has produced record-breaking perplexity results in several tasks (Mikolov et al., 2011a), and has the potential to encode sentence-span information in the network hidden-layer activations. We have also evaluated the use of parse scores, using an off-the-shelf stochastic context free grammar parser. How-

ever, the grammatical structure of the alternatives is often identical. With scores differing only in the final non-terminal/terminal rewrites, this did little better than chance. The use of other syntactically derived features, for example based on a dependency parse, are likely to be more effective, but we leave this for future work.

#### 3.1 Backoff N-gram Language Model

Our baseline model is a Good-Turing smoothed model trained with the CMU language modeling toolkit (Clarkson and Rosenfeld, 1997). For the SAT task, we used a trigram language model trained on 1.1B words of newspaper data, described in Section 5.1. All bigrams occurring at least twice were retained in the model, along with all trigrams occurring at least three times. The vocabulary consisted of all words occurring at least 100 times in the data, along with every word in the development or test sets. This resulted in a 124k word vocabulary and 59M n-grams. For the Conan Doyle data, which we henceforth refer to as the *Holmes data* (see Section 5.1), the smaller amount of training data allowed us to use 4-grams and a vocabulary cutoff of 3. This resulted in 26M n-grams and a 126k word vocabulary.

#### 3.2 Maximum Entropy Class-Based N-gram Language Model

Word-class information provides a level of abstraction which is not available in a word-level language model; therefore we evaluated a state-of-the-art class based language model. Model M (Chen, 2009a; Chen, 2009b) is a recently proposed class based exponential n-gram language model which has shown improvements across a variety of tasks (Chen, 2009b; Chen et al., 2009; Emami et al., 2010). The key ideas are the modeling of word n-gram probabilities with a maximum entropy model, and the use of word-class information in the definition of the features. In particular, each word  $w$  is assigned deterministically to a class  $c$ , allowing the n-gram probabilities to be estimated as the product of class and word parts

$$P(w_i | w_{i-n+1} \dots w_{i-2} w_{i-1}) = P(c_i | c_{i-n+1} \dots c_{i-2} c_{i-1}, w_{i-n+1} \dots w_{i-2} w_{i-1}) P(w_i | w_{i-n+1} \dots w_{i-2} w_{i-1}, c_i).$$

Both components are themselves maximum entropy n-gram models in which the probability of a word or class label  $l$  given history  $h$  is determined by  $\frac{1}{Z} \exp(\sum_k f_k(h, l))$ . The features  $f_k(h, l)$  used are the presence of various patterns in the concatenation of  $hl$ , for example whether a particular suffix is present in  $hl$ .

### 3.3 Recurrent Neural Net Language Model

Many of the questions involve long-range dependencies between words. While n-gram models have no ability to explicitly maintain long-span context, the recently proposed recurrent neural-net model of (Mikolov et al., 2010) does. Related approaches have been proposed by (Sutskever et al., 2011; Socher et al., 2011). In this model, a set of neural net activations  $\mathbf{s}(t)$  is maintained and updated at each sentence position  $t$ . These activations encapsulate the sentence history up to the  $t^{\text{th}}$  word in a real-valued vector which typically has several hundred dimensions. The word at position  $t$  is represented as a binary vector  $\mathbf{w}(t)$  whose length is the vocabulary size, and with a “1” in a position uniquely associated with the word, and “0” elsewhere.  $\mathbf{w}(t)$  and  $\mathbf{s}(t)$  are concatenated to predict an output distribution over words,  $\mathbf{y}(t)$ . Updating is done with two weight matrices  $\mathbf{u}$  and  $\mathbf{v}$  and nonlinear functions  $f()$  and  $g()$  (Mikolov et al., 2011b):

$$\begin{aligned} \mathbf{x}(t) &= [\mathbf{w}(t)^T \mathbf{s}(t-1)^T]^T \\ s_j(t) &= f\left(\sum_i x_i(t) u_{ji}\right) \\ y_k(t) &= g\left(\sum_j s_j(t) v_{kj}\right) \end{aligned}$$

with  $f()$  being a sigmoid and  $g()$  a softmax:

$$f(x) = \frac{1}{1 + \exp(-x)}, g(z_m) = \frac{\exp(z_m)}{\sum_k \exp(z_k)}$$

The output  $\mathbf{y}(t)$  is a probability distribution over words, and the parameters  $\mathbf{u}$  and  $\mathbf{v}$  are trained with back-propagation to minimize the Kullback-Leibler (KL) divergence between the predicted and observed distributions. Because of the recurrent connections, this model is similar to a nonlinear infinite impulse response (IIR) filter, and has the potential to model long span dependencies. Theoretical considerations (Bengio et al., 1994) indicate that for many problems, this may not be possible, but in practice it is an empirical question.

## 4 Sentence Completion via Latent Semantic Analysis

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) is a widely used method for representing words and documents in a low dimensional vector space. The method is based on applying singular value decomposition (SVD) to a matrix  $W$  representing the occurrence of words in documents. SVD results in an approximation of  $W$  by the product of three matrices, one in which each word is represented as a low-dimensional vector, one in which each document is represented as a low dimensional vector, and a diagonal scaling matrix. The similarity between two words can then be quantified as the cosine-similarity between their respective scaled vectors, and document similarity can be measured likewise. It has been used in numerous tasks, ranging from information retrieval (Deerwester et al., 1990) to speech recognition (Bellegarda, 2000; Coccaro and Jurafsky, 1998).

To perform LSA, one proceeds as follows. The input is a collection of  $n$  documents which are expressed in terms of words from a vocabulary of size  $m$ . These documents may be actual documents such as newspaper articles, or simply as in our case notional documents such as sentences. Next, a  $m \times n$  matrix  $W$  is formed. At its simplest, the  $ij^{\text{th}}$  entry contains the number of times word  $i$  has occurred in document  $j$  - its *term frequency* or TF value. More conventionally, the entry is weighted by some notion of the importance of word  $i$ , for example the negative logarithm of the fraction of documents that contain it, resulting in a TF-IDF weighting (Salton et al., 1975). Finally, to obtain a subspace representation of dimension  $d$ ,  $W$  is decomposed as

$$W \approx USV^T$$

where  $U$  is  $m \times d$ ,  $V^T$  is  $d \times n$ , and  $S$  is a  $d \times d$  diagonal matrix. In applications,  $d \ll n$  and  $d \ll m$ ; for example one might have a 50,000 word vocabulary and 1,000,000 documents and use a 300 dimensional subspace representation.

An important property of SVD is that the rows of  $US$  - which represents the words - behave similarly to the original rows of  $W$ , in the sense that the cosine similarity between two rows in  $US$  approximates the cosine similarity between the corre-



sponding rows in  $W$ . Cosine similarity is defined as  $\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}$ .

#### 4.1 Total Word Similarity

Perhaps the simplest way of doing sentence completion with LSA is to compute the total similarity of a potential answer  $a$  with the rest of the words in the sentence  $\mathcal{S}$ , and to choose the most related option. We define the total similarity as:

$$\text{tot sim}(a, \mathcal{S}) = \sum_{w \in \mathcal{S}} \text{sim}(a, w)$$

When the completion requires two words, total similarity is the sum of the contributions for both words. This is our baseline method for using LSA, and one of the best methods we have found.

#### 4.2 Sentence Reconstruction

Recall that LSA approximates a weighted word-document matrix  $W$  as the product of low rank matrices  $U$  and  $V$  along with a scaling matrix  $S$ :  $W \approx USV^T$ . Using singular value decomposition, this is done so as to minimize the mean square reconstruction error  $\sum_{ij} Q_{ij}^2$  where  $Q = W - USV^T$ . From the basic definition of LSA, each column of  $W$  (representing a document) is represented as

$$W_j = USV_j^T, \quad (1)$$

that is, as a linear combination of the set of basis functions formed by the columns of  $US$ , with the combination weights specified in  $V_j^T$ . When a new document is presented, it is also possible to represent it in terms of the same basis vectors. Moreover, we may take the reconstruction error induced by this representation to be a measure of how consistent the new document is with the original set of documents used to determine  $US$  and  $V$  (Bellegarda, 2000).

It remains to represent a new document in terms of the LSA bases. This is done as follows (Deerwester et al., 1990; Bellegarda, 2000), again with the objective of minimizing the reconstruction error. First, note that since  $U$  is column-orthonormal, (1) implies that

$$V_j = W_j^T US^{-1} \quad (2)$$

Thus, if we notionally index a new document by  $p$ , we proceed by forming a new column (document) vector  $W_p$  using the standard term-weighting, and

then find its LSA-space representation  $V_p$  using (2). We can evaluate the reconstruction quality by inserting the result in (1). The reconstruction error is then

$$\|(UU^T - I)W_p\|^2$$

Note that if all the dimensions are retained, the reconstruction error is zero; in the case that only the highest singular vectors are used, however, it is not. Due to the fact that the sentences vary in length we choose the number of retained singular vectors as a fraction  $f$  of the sentence length. If the answer has  $n$  words we use the top  $nf$  components. In practice, a  $f$  of 1.2 was selected on the basis of development set results.

#### 4.3 A LSA N-gram Language Model

In the context of speech recognition, LSA has been combined with classical n-gram language models in (Coccaro and Jurafsky, 1998; Bellegarda, 2000). The crux of this idea is to interpolate an n-gram language model probability with one based on LSA, with the intuition that the standard n-gram model will do a good job predicting function words, and the LSA model will do a good job on words predicted by their long-span context. This logic makes sense for the sentence completion task as well, motivating us to evaluate it.

To do this, we adopt the procedure of (Coccaro and Jurafsky, 1998), using linear interpolation between the n-gram and LSA probabilities:

$$p(w|history) = \alpha p_{ng}(w|history) + (1 - \alpha) p_{lsa}(w|history)$$

The probability of a word given its history is computed by the LSA model in the following way. Let  $h$  be the sum of all the LSA word vectors in the history. Let  $m$  be the smallest cosine similarity between  $h$  and any word in the vocabulary  $V$ :  $m = \min_{w \in V} \text{sim}(h, w)$ . The probability of a word  $w$  in the context of history  $h$  is given by

$$P_{lsa}(w|h) = \frac{\text{sim}(h, w) - m}{\sum_{q \in V} (\text{sim}(h, q) - m)}$$

Since similarity can be negative, subtracting the minimum ( $m$ ) ensures that all the estimated probabilities are between 0 and 1.

#### 4.4 Improving Efficiency and Expressiveness

Given the basic framework described above, a number of enhancements are possible. In terms of efficiency, recall that it is necessary to perform SVD on a term-document matrix. The data we used was grouped into paragraph “documents,” of which there were over 27 million, with 2.6 million unique words. While the resulting matrix is highly sparse, it is nevertheless impractical to perform SVD. We overcome this difficulty in two ways. First, we restrict the set of documents used to those which are “relevant” to a given test set. This is done by requiring that a document contain at least one of the potential answer-words. Secondly, we restrict the vocabulary to the set of words present in the test set. For the sentence-reconstruction method of Section 4.2, we have found it convenient to do data selection *per-sentence*.

To enhance the expressive power of LSA, the term vocabulary can be expanded from unigrams to bigrams or trigrams of words, thus adding information about word ordering. This was also used in the reconstruction technique.

### 5 Experimental Results

#### 5.1 Data Resources

We present results with two datasets. The first is taken from *11 Practice Tests for the SAT & PSAT 2011 Edition* (Princeton-Review, 2010). This book contains eleven practice tests, and we used all the sentence completion questions in the first five tests as a development set, and all the questions in the last six tests as the test set. This resulted in sets with 95 and 108 questions respectively. Additionally, we report results on the recently released *MSR Sentence Completion Challenge* (Zweig and Burges, 2011). This consists of a set of 1,040 sentence completion questions based on sentences occurring in five Conan Doyle Sherlock Holmes novels, and is identical in format to the SAT questions. Due to the source of this data, we refer to it as the *Holmes data*.

To train models, we have experimented with a variety of data sources. Since there is no publicly available collection of SAT questions suitable to training, our methods have all relied on unsupervised data. Early on, we ran a set of experiments to determine the relevance of different types of data. Thinking that data from an encyclopedia

Data	Dev % Correct	Test % Correct
Encarta	26	33
Wikipedia	32	31
LA Times	39	42

Table 1: Effectiveness of different types of training data.

might be useful, we evaluated an electronic version of the 2003 Encarta encyclopedia, which has approximately 29M words. Along similar lines, we used a collection of Wikipedia articles consisting of 709M words. This data is the entire Wikipedia as of January 2011, broken down into sentences, with filtering to remove sentences consisting of URLs and Wiki author comments. Finally, we used a commercial newspaper dataset consisting of all the Los Angeles Times data from 1985 to 2002, containing about 1.1B words. These data sources were evaluated using the baseline n-gram LM approach of Section 3.1. Initial experiments indicated that that the Los Angeles Times data is best suited to this task (see Table 1), and our SAT experiments use this source. For the MSR Sentence Completion data, we obtained the training data specified in (Zweig and Burges, 2011), consisting of approximately 500 19th-century novels available from Project Gutenberg, and comprising 48M words.

#### 5.2 Human Performance

To provide human benchmark performance, we asked six native speaking high school students and five graduate students to answer the questions on the development set. The high-schoolers attained 87% accuracy and the graduate students 95%. Zweig and Burges (2011) cite a human performance of 91% on the Holmes data. Statistics from a large cross-section of the population are not available. As a further point of comparison, we note that chance performance is 20%.

#### 5.3 Language Modeling Results

Table 2 summarizes our language modeling results on the SAT data. With the exception of the baseline backoff n-gram model, these techniques were too computationally expensive to utilize the full Los Angeles Times corpus. Instead, as with LSA, a “relevant” corpus was selected of the sentences which contain at least one answer option from either the

Method	Data (Dev / Test)	Dev	Test
3-gram GT	1.1B / 1.1B	39%	42%
Model M	193M / 236M	35	41
RNN	36M / 44M	37	42
LSA-LM	293M / 358 M	48	44

Table 2: Performance of language modeling methods on SAT questions.

Method	Dev ppl	Dev	Test ppl	Test
3-gram GT	195	36%	190	44%
Model M	178	36	175	42
RNN	147	37	144	42

Table 3: Performance of language modeling methods using identical training data and vocabularies.

development or test set. Separate subsets were made for development and test data. This data was further sub-sampled to obtain the training set sizes indicated in the second column. For the LSA-LM, an interpolation weight of 0.1 was used for the LSA score, determined through optimization on the development set. We see from this table that the language models perform similarly and achieve just above 40% on the test set.

To make a more controlled comparison that normalizes for the amount of training data, we have trained Model M, and the Good-Turing model on the same data subset as the RNN, and with the same vocabulary. In Table 3, we present perplexity results on a held-out set of dev/test-relevant Los Angeles Times data, and performance on the actual SAT questions. Two things are notable. First, the recurrent neural net has dramatically lower perplexity than the other methods. This is consistent with results in (Mikolov et al., 2011a). Secondly, despite the differences in perplexity, the methods show little difference on SAT performance. Because Model M was not better, only uses n-gram context, and was used in the construction of the Holmes data (Zweig and Burges, 2011), we do not consider it further.

## 5.4 LSA Results

Table 4 presents results for the methods of Sections 4.1 and 4.2. Of all the methods in isolation, the simple approach of Section 4.1 - to use the total cosine similarity between a potential answer and the other words in the sentence - has performed best. The ap-

Method	Dev	Test
Total Word Similarity	46%	46%
Reconstruction Error	53	41

Table 4: SAT performance of LSA based methods.

Method	Test
3-input LSA	46%
LSA + Good-Turing LM	53
LSA + Good-Turing LM + RNN	52

Table 5: SAT test set accuracy with combined methods.

proach of using reconstruction error performed very well on the development set, but unremarkably on the test set.

## 5.5 Combination Results

A well-known trick for obtaining best results from a machine learning system is to combine a set of diverse methods into a single ensemble (Dietterich, 2000). We use ensembles to get the highest accuracy on both of our data sets.

We use a simple linear combination of the outputs of the other models discussed in this paper. For the LSA model, the linear combination has three inputs: the total word similarity, the cosine similarity between the sum of the answer word vectors and the sum of the rest of sentence’s word vectors, and the number of out-of-vocabulary terms in the answer. Each additional language model beyond LSA contributes an additional input: the probability of the sentence under that language model.

We train the parameters of the linear combination on the SAT development set. The training minimizes a loss function of pairs of answers: one correct and one incorrect fill-in from the same question. We use the RankNet loss function (Burges et al., 2005):

$$\min_{\vec{w}} f(\vec{w} \cdot (\vec{x} - \vec{y})) + \lambda \|\vec{w}\|^2$$

where  $\vec{x}$  are the input features for the incorrect answer,  $\vec{y}$  are the features for the correct answer,  $\vec{w}$  are the weights for the combination, and  $f(z) = \log(1 + \exp(z))$ . We tune the regularizer via 5-fold cross validation, and minimize the loss using L-BFGS (Nocedal and Wright, 2006). The results on the SAT test set for combining various models are shown in Table 5.

## 5.6 Holmes Data Results

To measure the robustness of our approaches, we have applied them to the MSR Sentence Completion set (Zweig and Burges, 2011), termed the *Holmes data*. In Table 6, we present the results on this set, along with the comparable SAT results. Note that the latter are derived from models trained with the Los Angeles Times data, while the Holmes results are derived from models trained with 19th-century novels. We see from this table that the results are similar across the two tasks. The best performing single model is LSA total word similarity.

For the Holmes data, combining the models outperforms any single model. We train the linear combination function via 5-fold cross-validation: the model is trained five times, each time on 3/5 of the data, the regularization tuned on 1/5 of the data, and tested on 1/5. The test results are pooled across all 5 folds and are shown in Table 6. In this case, the best combination is to blend LSA, the Good-Turing language model, and the recurrent neural network.

## 6 Discussion

To verify that the differences in accuracy between the different algorithms are not statistical flukes, we perform a statistical significance test on the outputs of each algorithm. We use McNemar’s test, which is a matched test between two classifiers (Dietterich, 1998). We use the False Discovery Rate method (Benjamini and Hochberg, 1995) to control the false positive rate caused by multiple tests. If we allow 2% of our tests to yield incorrectly false results, then for the SAT data, the combination of the Good-Turing smoothed language model with an LSA-based global similarity model (52% accuracy) is better than the baseline alone (42% accuracy).

Secondly, for the Holmes data, we can state that LSA total similarity beats the recurrent neural network, which in turn is better than the baseline n-gram model. The combination of all three is significantly better than any of the individual models.

To better understand the system performance and gain insight into ways of improving it, we have examined the system’s errors. Encouragingly, one-third of the errors involve single-word questions which test the dictionary definition of a word. This is done either by stating the definition, or provid-

Method	SAT	Holmes
Chance	20%	20%
GT N-gram LM	42	39
RNN	42	45
LSA Total Similarity	46	49
Reconstruction Error	41	41
LSA-LM	44	42
Combination	<b>53</b>	<b>52</b>
Human	87 to 95	91

Table 6: Performance of methods on the MSR Sentence Completion Challenge, contrasted with SAT test set.

ing a stereotypical use of the word. An example of the first case is: “Great artists are often *prophetic (visual)*: they perceive what we cannot and anticipate the future long before we do.” (The system’s incorrect answer is in parentheses.) An example of the second is: “One cannot help but be moved by Theresa’s *heartrending (therapeutic)* struggle to overcome a devastating and debilitating accident.”

At the other end of the difficulty spectrum are questions involving world knowledge and/or logical implications. An example requiring both is, “Many fear that the *ratification (withdrawal)* of more lenient tobacco advertising could be detrimental to public health.” About 40% of the errors require this sort of general knowledge to resolve. Based on our analysis, we believe that future research could profitably exploit the structured information present in a dictionary. However, the ability to identify and manipulate logical relationships and embed world knowledge in a manner amenable to logical manipulation may be necessary for a full solution. It is an interesting research question if this could be done implicitly with a machine learning technique, for example recurrent or recursive neural networks.

## 7 Conclusion

In this paper we have investigated methods for answering sentence-completion questions. These questions are intriguing because they probe the ability to distinguish semantically coherent sentences from incoherent ones, and yet involve no more context than the single sentence. We find that both local n-gram information and an LSA-based global coherence model do significantly better than chance, and that they can be effectively combined.

## References

- J. Bellegarda. 2000. Exploiting latent semantic information in statistical language modeling. *Proceedings of the IEEE*, 88(8).
- Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.
- Y. Benjamini and Y. Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. Royal Statistical Society B*, 53(1):289–300.
- C. Burges, T. Shaked., E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. 2005. Learning to rank using gradient descent. In *Proc. ICML*, pages 89–96.
- Eugene Charniak, Yasemin Altun, Rodrigo de Salvo Braz, Benjamin Garrett, Margaret Kosmala, Tomer Moscovich, Lixin Pang, Changhee Pyo, Ye Sun, Wei Wy, Zhongfa Yang, Shawn Zeller, and Lisa Zorn. 2000. Reading comprehension programs in a statistical-language-processing class. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading comprehension tests as evaluation for computer-based language understanding systems - Volume 6*, ANLP/NAACL-ReadingComp '00, pages 1–5. Association for Computational Linguistics.
- Stanley Chen and Joshua Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–393.
- S. Chen, L. Mangu, B. Ramabhadran, R. Sarikaya, and A. Sethy. 2009. Scaling shrinkage-based language models. In *ASRU*.
- S. Chen. 2009a. Performance prediction for exponential language models. In *NAACL-HLT*.
- S. Chen. 2009b. Shrinking exponential language models. In *NAACL-HLT*.
- P.R. Clarkson and R. Rosenfeld. 1997. Statistical language modeling using the CMU-Cambridge Toolkit. In *Proceedings ESCA Eurospeech*, <http://www.speech.cs.cmu.edu/SLM/toolkit.html>.
- N. Coccaro and D. Jurafsky. 1998. Towards better integration of semantic predictors in statistical language modeling. In *Proceedings, ICSLP*.
- Bollegala D., Matsuo Y., and Ishizuka M. 2009. Measuring the similarity between implicit semantic relations from the web. In *World Wide Web Conference (WWW)*.
- S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(96).
- T.G. Dietterich. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10:1895–1923.
- T.G. Dietterich. 2000. Ensemble methods in machine learning. In *International Workshop on Multiple Classifier Systems*, pages 1–15. Springer-Verlag.
- Educational-Testing-Service. 2011. [https://satonlinecourse.collegeboard.com/sr/digital\\_assets/assessment/pdf/0833a611-0a43-10c2-0148-cc8c0087fb06-f.pdf](https://satonlinecourse.collegeboard.com/sr/digital_assets/assessment/pdf/0833a611-0a43-10c2-0148-cc8c0087fb06-f.pdf).
- A. Emami, S. Chen, A. Ittycheriah, H. Soltau, and B. Zhao. 2010. Decoding with shrinkage-based language models. In *Interspeech*.
- Claudio Giuliano, Alfio Gliozzo, and Carlo Strapparava. 2007. Fbk-irst: Lexical substitution task exploiting domain and syntagmatic coherence. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 145–148, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Samer Hassan, Andras Csomai, Carmen Banea, Ravi Sinha, and Rada Mihalcea. 2007. Unt: Subfinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, SemEval '07, pages 410–413, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lynette Hirschman, Mark Light, Eric Breck, and John D. Burger. 1999. Deep read: A reading comprehension system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*.
- Thomas Landauer and Susan Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2), pages 211–240.
- Iddo Lev, Bill MacCartney, Christopher D. Manning, and Roger Levy. 2004. Solving logic puzzles: from robust processing to precise semantics. In *Proceedings of the 2nd Workshop on Text Meaning and Interpretation*, pages 9–16. Association for Computational Linguistics.
- Jarmasz M. and Szpakowicz S. 2003. Roget's thesaurus and semantic similarity. In *Recent Advances in Natural Language Processing (RANLP)*.
- Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53.
- Tomas Mikolov, Martin Karafiat, Jan Cernocky, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of Interspeech 2010*.

- Tomas Mikolov, Anoop Deoras, Stefan Kombrink, Lukas Burget, and Jan Cernocky. 2011a. Empirical evaluation and combination of advanced language modeling techniques. In *Proceedings of Interspeech 2011*.
- Tomas Mikolov, Stefan Kombrink, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur. 2011b. Extensions of recurrent neural network based language model. In *Proceedings of ICASSP 2011*.
- Saif Mohammed, Bonnie Dorr, and Graeme Hirst. 2008. Computing word pair antonymy. In *Empirical Methods in Natural Language Processing (EMNLP)*.
- Saif M. Mohammed, Bonnie J. Dorr, Graeme Hirst, and Peter D. Turney. 2011. Measuring degrees of semantic opposition. Technical report, National Research Council Canada.
- Hwee Tou Ng, Leong Hwee Teo, and Jennifer Lai Pheng Kwan. 2000. A machine learning approach to answering questions for reading comprehension tests. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics - Volume 13*, EMNLP '00, pages 124–132.
- J. Nocedal and S. Wright. 2006. *Numerical Optimization*. Springer-Verlag.
- Sebastian Pado and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33 (2), pages 161–199.
- Princeton-Review. 2010. *11 Practice Tests for the SAT & PSAT, 2011 Edition*. The Princeton Review.
- Ellen Riloff and Michael Thelen. 2000. A rule-based question answering system for reading comprehension tests. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading comprehension tests as evaluation for computer-based language understanding systems - Volume 6*, ANLP/NAACL-ReadingComp '00, pages 13–19.
- G. Salton, A. Wong, and C. S. Yang. 1975. A Vector Space Model for Automatic Indexing. *Communications of the ACM*, 18(11).
- Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 2011 International Conference on Machine Learning (ICML-2011)*.
- Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 2011 International Conference on Machine Learning (ICML-2011)*.
- E. Terra and C. Clarke. 2003. Frequency estimates for statistical word similarity measures. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.
- Peter Turney and Michael Littman. 2005. Corpus-based learning of analogies and semantic relations. *Machine Learning*, 60 (1-3), pages 251–278.
- Peter D. Turney, Michael L. Littman, Jeffrey Bigham, and Victor Shnayder. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. In *Recent Advances in Natural Language Processing (RANLP)*.
- Peter D. Turney. 2001. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *European Conference on Machine Learning (ECML)*.
- Peter Turney. 2008. A uniform approach to analogies, synonyms, antonyms, and associations. In *International Conference on Computational Linguistics (COLING)*.
- T. Veale. 2004. Wordnet sits the sat: A knowledge-based approach to lexical analogy. In *European Conference on Artificial Intelligence (ECAI)*.
- W. Wang, J. Auer, R. Parasuraman, I. Zubarev, D. Brandyberry, and M. P. Harper. 2000. A question answering system developed as a project in a natural language processing course. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading comprehension tests as evaluation for computer-based language understanding systems - Volume 6*, ANLP/NAACL-ReadingComp '00, pages 28–35.
- Deniz Yuret. 2007. Ku: word sense disambiguation by substitution. In *Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval '07*, pages 207–213, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Geoffrey Zweig and Christopher J.C. Burges. 2011. The Microsoft Research sentence completion challenge. Technical Report MSR-TR-2011-129, Microsoft.

# Iterative Viterbi A\* Algorithm for $K$ -Best Sequential Decoding

Zhiheng Huang<sup>†</sup>, Yi Chang, Bo Long, Jean-Francois Crespo<sup>†</sup>,  
Anlei Dong, Sathiya Keerthi and Su-Lin Wu

Yahoo! Labs

701 First Avenue, Sunnyvale  
CA 94089, USA

{zhiheng\_huang, jfcrespo}@yahoo.com<sup>†</sup>

{yichang, bolong, anlei, selvarak, sulin}@yahoo-inc.com

## Abstract

Sequential modeling has been widely used in a variety of important applications including named entity recognition and shallow parsing. However, as more and more real time large-scale tagging applications arise, decoding speed has become a bottleneck for existing sequential tagging algorithms. In this paper we propose 1-best A\*, 1-best iterative A\*,  $k$ -best A\* and  $k$ -best iterative Viterbi A\* algorithms for sequential decoding. We show the efficiency of these proposed algorithms for five NLP tagging tasks. In particular, we show that iterative Viterbi A\* decoding can be several times or orders of magnitude faster than the state-of-the-art algorithm for tagging tasks with a large number of labels. This algorithm makes real-time large-scale tagging applications with thousands of labels feasible.

## 1 Introduction

Sequence tagging algorithms including HMMs (Rabiner, 1989), CRFs (Lafferty et al., 2001), and Collins's perceptron (Collins, 2002) have been widely employed in NLP applications. Sequential decoding, which finds the best tag sequences for given inputs, is an important part of the sequential tagging framework. Traditionally, the Viterbi algorithm (Viterbi, 1967) is used. This algorithm is quite efficient when the label size of problem modeled is low. Unfortunately, due to its  $O(TL^2)$  time complexity, where  $T$  is the input token size and  $L$  is the label size, the Viterbi decoding can become prohibitively slow when the label size is large (say, larger than 200).

It is not uncommon that the problem modeled consists of more than 200 labels. The Viterbi algorithm cannot find the best sequences in tolerable

response time. To resolve this, Esposito and Radicioni (2009) have proposed a Carpediem algorithm which opens only necessary nodes in searching the best sequence. More recently, Kaji et al. (2010) proposed a staggered decoding algorithm, which proves to be very efficient on datasets with a large number of labels.

What the aforementioned literature does not cover is the  $k$ -best sequential decoding problem, which is indeed frequently required in practice. For example to pursue a high recall ratio, a named entity recognition system may have to adopt  $k$ -best sequences in case the true entities are not recognized at the best one. The  $k$ -best parses have been extensively studied in syntactic parsing context (Huang, 2005; Pauls and Klein, 2009), but it is not well accommodated in sequential decoding context. To our best knowledge, the state-of-the-art  $k$ -best sequential decoding algorithm is *Viterbi A\**<sup>1</sup>. In this paper, we generalize the *iterative process* from the work of (Kaji et al., 2010) and propose a  $k$ -best sequential decoding algorithm, namely *iterative Viterbi A\**. We show that the proposed algorithm is several times or orders of magnitude faster than the state-of-the-art in all tagging tasks which consist of more than 200 labels.

Our contributions can be summarized as follows. (1) We apply the A\* search framework to sequential decoding problem. We show that A\* with a proper heuristic can outperform the classic Viterbi decoding. (2) We propose 1-best A\*, 1-best iterative A\* decoding algorithms which are the second and third fastest decoding algorithms among the five decoding algorithms for comparison, although there is a significant gap to the fastest 1-best decoding algorithm. (3) We propose  $k$ -best A\* and  $k$ -best iterative Viterbi A\* algorithms. The latter is several times or orders of magnitude faster than the state-of-the-art

<sup>1</sup>Implemented in both CRFPP (<http://crfpp.sourceforge.net/>) and LingPipe (<http://alias-i.com/lingpipe/>) packages.

$k$ -best decoding algorithm. This algorithm makes real-time large-scale tagging applications with thousands of labels feasible.

## 2 Problem formulation

In this section, we formulate the sequential decoding problem in the context of perceptron algorithm (Collins, 2002) and CRFs (Lafferty et al., 2001). All the discussions apply to HMMs as well. Formally, a perceptron model is

$$f(\mathbf{y}, \mathbf{x}) = \sum_{t=1}^T \sum_{k=1}^K \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t), \quad (1)$$

and a CRFs model is

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{\sum_{t=1}^T \sum_{k=1}^K \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\}, \quad (2)$$

where  $\mathbf{x}$  and  $\mathbf{y}$  is an observation sequence and a label sequence respectively,  $t$  is the sequence position,  $T$  is the sequence size,  $f_k$  are feature functions and  $K$  is the number of feature functions.  $\theta_k$  are the parameters that need to be estimated. They represent the importance of feature functions  $f_k$  in prediction. For CRFs,  $Z(\mathbf{x})$  is an instance-specific normalization function

$$Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left\{\sum_{t=1}^T \sum_{k=1}^K \theta_k f_k(y_t, y_{t-1}, \mathbf{x}_t)\right\}. \quad (3)$$

If  $\mathbf{x}$  is given, the decoding is to find the best  $\mathbf{y}$  which maximizes the score of  $f(\mathbf{y}, \mathbf{x})$  for perceptron or the probability of  $p(\mathbf{y}|\mathbf{x})$  for CRFs. As  $Z(\mathbf{x})$  is a constant for any given input sequence  $\mathbf{x}$ , the decoding for perceptron or CRFs is identical, that is,

$$\arg \max_{\mathbf{y}} f(\mathbf{y}, \mathbf{x}). \quad (4)$$

To simplify the discussion, we divide the features into two groups: unigram label features and bigram label features. Unigram features are of form  $f_k(y_t, \mathbf{x}_t)$  which are concerned with the current label and arbitrary feature patterns from input sequence. Bigram features are of form  $f_k(y_t, y_{t-1}, \mathbf{x}_t)$  which are concerned with both the previous and the current labels. We thus rewrite the decoding problem as

$$\arg \max_{\mathbf{y}} \sum_{t=1}^T \left( \sum_{k=1}^{K_1} \theta_k^1 f_k^1(y_t, \mathbf{x}_t) + \sum_{k=1}^{K_2} \theta_k^2 f_k^2(y_t, y_{t-1}, \mathbf{x}_t) \right). \quad (5)$$

For a better understanding, one can interpret the term  $\sum_{k=1}^{K_1} \theta_k^1 f_k^1(y_t, \mathbf{x}_t)$  as node  $y_t$ 's score at position  $t$ , and interpret the term

$\sum_{k=1}^{K_2} \theta_k^2 f_k^2(y_t, y_{t-1}, \mathbf{x}_t)$  as edge  $(y_{t-1}, y_t)$ 's score. So the sequential decoding problem is cast as a max score pathfinding problem<sup>2</sup>. In the discussion hereafter, we assume scores of nodes and edges are pre-computed (denoted as  $n(y_t)$  and  $e(y_{t-1}, y_t)$ ), and we can thus focus on the analysis of different decoding algorithms.

## 3 Background

We present the existing algorithms for both 1-best and  $k$ -best sequential decoding in this section. These algorithms serve as basis for the proposed algorithms in Section 4.

### 3.1 1-Best Viterbi

The Viterbi algorithm is a classic dynamic programming based decoding algorithm. It has the computational complexity of  $O(TL^2)$ , where  $T$  is the input sequence size and  $L$  is the label size<sup>3</sup>. Formally, the Viterbi computes  $\alpha(y_t)$ , the best score from starting position to label  $y_t$ , as follows.

$$\max_{y_{t-1}} (\alpha_{y_{t-1}} + e(y_{t-1}, y_t)) + n(y_t), \quad (6)$$

where  $e(y_{t-1}, y_t)$  is the edge score between nodes  $y_{t-1}$  and  $y_t$ ,  $n(y_t)$  is the node score for  $y_t$ . Note that the terms  $\alpha_{y_{t-1}}$  and  $e(y_{t-1}, y_t)$  take value 0 for  $t = 0$  at initialization. Using the recursion defined above, we can compute the highest score at end position  $T - 1$  and its corresponding sequence. The recursive computation of  $\alpha_{y_t}$  is denoted as forward pass since the computing traverses the lattice from left to right. Conversely, the backward pass computes  $\beta_{y_t}$  as the follows.

$$\max_{y_{t+1}} (\beta_{y_{t+1}} + e(y_t, y_{t+1}) + n(y_{t+1})). \quad (7)$$

Note that  $\beta_{y_{T-1}} = 0$  at initialization. The max score can be computed using  $\max_{y_0} (\beta_0 + n(y_0))$ . We can use either forward or backward pass to compute the best sequence. Table 1 summarizes the computational complexity of all decoding algorithms including Viterbi, which has the complexity of  $TL^2$  for both best and worst cases. Note that N/A means the decoding algorithms are not applicable (for example, iterative Viterbi is not applicable to  $k$ -best decoding). The proposed algorithms (see Section 4) are highlighted in bold.

### 3.2 1-Best iterative Viterbi

Kaji et al. (Kaji et al., 2010) presented an efficient sequential decoding algorithm named *staggered decoding*. We use the name *iterative Viterbi* to describe

<sup>2</sup>With the constraint that the path consists of one and only one node at each position.

<sup>3</sup>We ignore the feature size terms for simplicity.



this algorithm for the reason that the iterative process plays a central role in this algorithm. Indeed, this iterative process is generalized in this paper to handle  $k$ -best sequential decoding (see Section 4.4).

The main idea is to start with a coarse lattice which consists of both *active* labels and *degenerate* labels. A label is referred to as an *active label* if it is not grouped (e.g., all labels in Fig. 1 (a) and label *A* at each position in Fig. 1 (b)), and otherwise as an *inactive label* (i.e., dotted nodes). The new label, which is made by grouping the inactive labels, is referred to as a *degenerate label* (i.e., large nodes covering the dotted ones). Fig. 1 (a) shows a lattice which consists of active labels only and (b) shows a lattice which consists of both active and degenerate ones. The score of a degenerate label is the max score of inactive labels which are included in the degenerate label. Similarly, the edge score between a degenerate label  $z$  and an active label  $y'$  is the max edge score between any inactive label  $y \in z$  and  $y'$ , and the score of two degenerate labels  $z$  and  $z'$  is the max edge score between any inactive label  $y \in z$  and  $y' \in z'$ . Using the above definitions, the best sequence derived from a degenerate lattice would be the upper bound of the sequence derived from the original lattice. If the best sequence does not include any degenerate labels, it is indeed the best sequence for the original lattice.

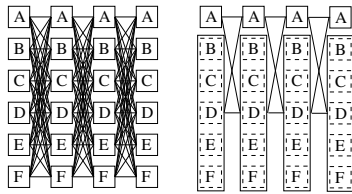


Figure 1: (a) A lattice consisting of active labels only. (b) A lattice consisting of both active labels and degenerate ones. Each position has one active label (A) and one degenerate label (consisting of B, C, D, E, and F).

The pseudo code for this algorithm is shown in Algorithm 1. The lattice is initialized to include one active label and one degenerate label at each position (see Figure 1 (b)). Note that the labels are ranked by the probabilities estimated from the training data. The Viterbi algorithm is applied to the lattice to find the best sequence. If the sequence consists of active labels only, the algorithm terminates and returns such a sequence. Otherwise, the lower bound  $lb^4$  of the active sequence in the lattice is updated and the lattice is expanded. The lower bound can be initialized to the best sequence score using a beam search (with beam size being 1). After either a forward or a backward pass, the lower bound is assigned with

<sup>4</sup>The maximum score of the active sequences found so far.

the best active sequence score  $best(lattice)^5$  if the former is less than the latter. The expansion of lattice ensures that the lattice has twice active labels as before at a given position. Figure 2 shows the column-wise expansion step. The number of active labels in the column is doubled only if the best sequence of the degenerate lattice passes through the degenerate label of that column.

---

### Algorithm 1 Iterative Viterbi Algorithm

---

```

1:  $lb =$  best score from beam search
2: init lattice
3: for  $i=0; i < T; i++$  do
4:   if  $i \% 2 == 0$  then
5:      $y =$  forward()
6:   else
7:      $y =$  backward()
8:   end if
9:   if  $y$  consists of active labels only then
10:    return  $y$ 
11:   end if
12:   if  $lb < best(lattice)$  then
13:      $lb = best(lattice)$ 
14:   end if
15:   expand lattice
16: end for

```

---



---

### Algorithm 2 Forward

---

```

1: for  $i=0; i < T; i++$  do
2:   Compute  $\alpha(y_i)$  and  $\beta(y_i)$  according to Equations (6) and (7)
3:   if  $\alpha(y_i) + \beta(y_i) < lb$  then
4:     prune  $y_i$  from the current lattice
5:   end if
6: end for
7:  $Node\ b = \arg \max_{y_{T-1}} \alpha(y_{T-1})$ 
8: return sequence back tracked by  $b$ 

```

---

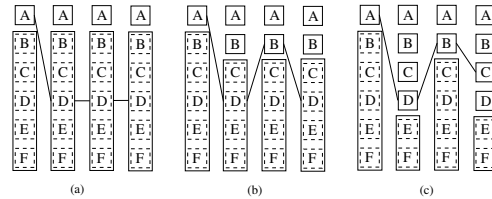


Figure 2: Column-wise lattice expansion: (a) The best sequence of the initial degenerate lattice, which does not pass through the degenerate label in the first column. (b) Column-wise expansion is performed and the best sequence is searched again. Notice that the active label in the first column is not expanded. (c) The final result.

Algorithm 2 shows the forward pass in which the node pruning is performed. That is, for any node, if the best score of sequence which passes such a node is less than the lower bound  $lb$ , such a node is removed from the lattice. This removal is safe as such a node does not have a chance to form an optimal sequence. It is worth noting that, if a node is removed, it can no longer be added into the lattice.

<sup>5</sup>We do not update the lower bound  $lb$  if we cannot find an active sequence.

This property ensures the efficiency of the iterative Viterbi algorithm. The backward pass is similar to the forward one and it is thus omitted.

The alternative calls of forward and backward passes (in Algorithm 1) ensure the alternative updating/lowering of node forward and backward scores, which makes the node pruning in either forward pass (see Algorithm 2) or backward pass more efficient. The lower bound  $lb$  is updated once in each iteration of the main loop in Algorithm 1. While the forward and backwards scores of nodes gradually decrease and the lower bound  $lb$  increases, more and more nodes are pruned.

The iterative Viterbi algorithm has computational complexity of  $T$  and  $TL^2$  for best and worst cases respectively. This can be proved as follows (Kaji et al., 2010). At the  $m$ -th iteration in Algorithm 1, iterative Viterbi decoding requires order of  $T4^m$  time because there are  $2^m$  active labels (plus one degenerate label). Therefore, it has  $\sum_{i=0}^m T4^i$  time complexity if it terminates at the  $m$ -th iteration. In the best case in which  $m = 0$ , the time complexity is  $T$ . In the worst case in which  $m = \lceil \log_2 L \rceil - 1$  ( $\lceil \cdot \rceil$  is the ceiling function which maps a real number to the smallest following integer), the time complexity is order of  $TL^2$  because  $\sum_{i=0}^{\lceil \log_2 L \rceil - 1} T4^i < 4/3TL^2$ .

### 3.3 1-Best Carpediem

Esposito and Radicioni (2009) have proposed a novel 1-best<sup>6</sup> sequential decoding algorithm, *Carpediem*, which attempts to open only necessary nodes in searching the best sequence in a given lattice. Carpediem has the complexity of  $TL \log L$  and  $TL^2$  for the best and worst cases respectively. We skip the description of this algorithm due to space limitations. Carpediem is used as a baseline in our experiments for decoding speed comparison.

### 3.4 K-Best Viterbi

In order to produce  $k$ -best sequences, it is not enough to store 1-best label per node, as the  $k$ -best sequences may include suboptimal labels. The  $k$ -best sequential decoding gives up this 1-best label memorization in the dynamic programming paradigm. It stores up to  $k$ -best labels which are necessary to form  $k$ -best sequences. The  $k$ -best Viterbi algorithm thus has the computational complexity of  $KTL^2$  for both best and worst cases.

Once we store the  $k$ -best labels per node in a lattice, the  $k$ -best Viterbi algorithm calls either the forward or the backward passes just in the same way as the 1-best Viterbi decoding does. We can compute

the  $k$  highest score at the end position  $T - 1$  and the corresponding  $k$ -best sequences.

### 3.5 K-Best Viterbi A\*

To our best knowledge the most efficient  $k$ -best sequence algorithm is the Viterbi A\* algorithm as shown in Algorithm 3. The algorithm consists of one forward pass and an A\* backward pass. The forward pass computes and stores the Viterbi forward scores, which are the best scores from the start to the current nodes. In addition, each node stores a *backlink* which points to its predecessor.

The major part of Algorithm 3 describes the backward A\* pass. Before describing the algorithm, we note that each node in the agenda represents a sequence. So the operations on nodes (push or pop) correspond to the operations on sequences. Initially, the  $L$  nodes at position  $T - 1$  are pushed to an agenda. Each of the  $L$  nodes  $n_i$ ,  $i = 0, \dots, L - 1$ , represents a sequence. That is, node  $n_i$  represents the best sequence from the start to itself. The best of the  $L$  sequences is the globally best sequence. However, the  $i$ -th best,  $i = 2, \dots, k$ , of the  $L$  sequence may not be the globally  $i$ -th best sequence. The priority of each node is set as the score of the sequence which is derived by such a node. The algorithm then goes to a loop of  $k$ . In each loop, the best node is popped off from the agenda and is stored in a set  $r$ . The algorithm adds alternative candidate nodes (or sequences) to the agenda via a double nested loop. The idea is that, when an optimal node (or sequence) is popped off, we have to push to the agenda all nodes (sequences) which are slightly worse than the just popped one. The interpretation of slightly worse is to replace one edge from the popped node (sequence). The slightly worse sequences can be found by the exact heuristic derived from the first Viterbi forward pass.

Figure 3 shows an example of the push operations for a lattice of  $T = 4$ ,  $Y = 4$ . Suppose an optimal node  $2:B$  (in red, standing for node  $B$  at position 2, representing the sequence of  $0:A 1:D 2:B 3:C$ ) is popped off, new nodes of  $1:A$ ,  $1:B$ ,  $1:C$  and  $0:B$ ,  $0:C$  and  $0:D$  are pushed to the agenda according to the double nested for loop in Algorithm 3. Each of the pushed nodes represents a sequence, for example, node  $1:B$  represents a sequence which consists of three parts: Viterbi sequence from start to  $1:B$  ( $0:C 1:B$ ),  $2:B$  and forward link of  $2:B$  ( $3:C$  in this case). All of these pushed nodes (sequences) are served as candidates for the next agenda pop operation.

The algorithm terminates the loop once it has optimal  $k$  nodes. The  $k$ -best sequences can be derived by the  $k$  optimal nodes. This algorithm has

<sup>6</sup>They did not provide  $k$ -best solutions.

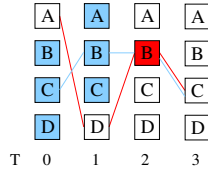


Figure 3: Alternative nodes push after popping an optimal node.

computation complexity of  $TL^2 + TL$  for both best and worst cases, with the first term accounting for Viterbi forward pass and the second term accounting for A\* backward process. The bottleneck is thus at the Viterbi forward pass.

---

**Algorithm 3** *K*-Best Viterbi A\* algorithm

---

```

1: forward()
2: push  $L$  best nodes to agenda  $q$ 
3:  $c = 0$ 
4:  $r = \{\}$ 
5: while  $c < K$  do
6:   Node  $n = q.pop()$ 
7:    $r = r \cup n$ 
8:   for  $i = n.t - 1; i \geq 0; i --$  do
9:     for  $j = 0; j < L; j ++$  do
10:      if  $j! = n.backlink.y$  then
11:        create new node  $s$  at position  $i$  and label  $j$ 
12:         $s.forwardlink = n$ 
13:         $q.push(s)$ 
14:      end if
15:    end for
16:     $n = n.backlink$ 
17:  end for
18:   $c ++$ 
19: end while
20: return  $K$  best sequences derived by  $r$ 

```

---

## 4 Proposed Algorithms

In this section, we propose A\* based sequential decoding algorithms that can efficiently handle datasets with a large number of labels. In particular, we first propose the A\* and the iterative A\* decoding algorithm for 1-best sequential decoding. We then extend the 1-best A\* algorithm to a  $k$ -best A\* decoding algorithm. We finally apply the iterative process to the Viterbi A\* algorithm, resulting in the *iterative Viterbi A\** decoding algorithm.

### 4.1 1-Best A\*

A\*(Hart et al., 1968; Russell and Norvig, 1995), as a classic search algorithm, has been successfully applied in syntactic parsing (Klein and Manning, 2003; Pauls and Klein, 2009). The general idea of A\* is to consider labels  $y_t$  which are likely to result in the best sequence using a score  $f$  as follows.

$$f(y) = g(y) + h(y), \quad (8)$$

where  $g(y)$  is the score from start to the current node and  $h(y)$  is a heuristic which estimates the score

from the current node to the target. A\* uses an agenda (based on the  $f$  score) to decide which nodes are to be processed next. If the heuristic satisfies the condition  $h(y_{t-1}) \geq e(y_{t-1}, y_t) + h(y_t)$ , then  $h$  is called *monotone* or *admissible*. In such a case, A\* is guaranteed to find the best sequence. We start with the naive (but admissible) heuristic as follows

$$h(y_t) = \sum_{i=t+1}^{T-1} (\max n(y_i) + \max e(y_{i-1}, y_i)). \quad (9)$$

That is, the heuristic of node  $y_t$  to the end is the sum of max edge scores between any two positions and max node scores per position. Similar to (Pauls and Klein, 2009) we explore the heuristic in different coarse levels. We apply the Viterbi backward pass to different degenerate lattices and use the Viterbi backward scores as different heuristics. Different degenerate lattices are generated from different iterations of Algorithm 1: The  $m$ -th iteration corresponds to a lattice of  $(2^m + 1) * T$  nodes. A larger  $m$  indicates a more accurate heuristic, which results in a more efficient A\* search (fewer nodes being processed). However, this efficiency comes with the price that such an accurate heuristic requires more computation time in the Viterbi backward pass. In our experiments, we try the naive heuristic and the following values of  $m$ : 0, 3, 6 and 9.

In the best case, A\* expands one node per position, and each expansion results in the push of all nodes at next position to the agenda. The search is similar to the beam search with beam size being 1. The complexity is thus  $TL$ . In the worst case, A\* expands every node per position, and each expansion results in the push of all nodes at next position to the agenda. The complexity thus becomes  $TL^2$ .

### 4.2 1-Best Iterative A\*

The iterative process as described in the iterative Viterbi decoding can be used to boost A\* algorithm, resulting in the *iterative A\** algorithm. For simplicity, we only make use of the naive heuristic in Equation (9) in the iterative A\* algorithm. We initialize the lattice with one active label and one degenerate label at each position (see Figure 1 (b)). We then run A\* algorithm on the degenerate lattice and get the best sequence. If the sequence is active we return it. Otherwise we expand the lattice in each iteration until we find the best active sequence. Similar to iterative Viterbi algorithm, iterative A\* has the complexity of  $T$  and  $TL^2$  for the best and worst cases respectively.

### 4.3 $K$ -Best A\*

The extension from 1-best A\* to  $k$ -best A\* is again due to the memorization of  $k$ -best labels per node.

Table 1: Best case and worst case computational complexity of various decoding algorithms.

	1-best decoding		$K$ -best decoding	
	best case	worst case	best case	worst case
beam	$TL$	$TL$	$KTL$	$KTL$
Viterbi	$TL^2$	$TL^2$	$KTL^2$	$KTL^2$
iterative Viterbi	$T$	$TL^2$	N/A	N/A
Carpediem	$TL \log L$	$TL^2$	N/A	N/A
A*	$TL$	$TL^2$	$KTL$	$KTL^2$
<b>iterative A*</b>	$T$	$TL^2$	N/A	N/A
Viterbi A*	N/A	N/A	$TL^2 + KTL$	$TL^2 + KTL$
<b>iterative Viterbi A*</b>	N/A	N/A	$T + KT$	$TL^2 + KTL$

We use either the naive heuristic (Equation (9)) or different coarse level heuristics by setting  $m$  to be 0, 3, 6 or 9 (see Section 4.1). The first  $k$  nodes which are popped off the agenda can be used to back track the  $k$ -best sequences. The  $k$ -best A\* algorithm has the computational complexity of  $KTL$  and  $KTL^2$  for best and worst cases respectively.

#### 4.4 $K$ -Best Iterative Viterbi A\*

We now present the  $k$ -best iterative Viterbi A\* algorithm (see Algorithm 4) which applies the iterative process to  $k$ -best Viterbi A\* algorithm. The major difference between 1-best iterative Viterbi A\* algorithm (Algorithm 1) and this algorithm is that the latter calls the  $k$ -best Viterbi A\* (Algorithm 3) after the best sequence is found. If the  $k$ -best sequences are all active, we terminate the algorithm and return the  $k$ -best sequences. If we cannot find either the best active sequence or the  $k$ -best active sequences, we expand the lattice to continue the search in the next iteration.

As in the iterative Viterbi algorithm (see Section 3.2), nodes are pruned at each position in forward or backward passes. Efficient pruning contributes significantly to speeding up decoding. Therefore, to have a tighter (higher) lower bound  $lb$  is important. We initialize the lower bound  $lb$  with the  $k$ -th best score from beam search (with beam size being  $k$ ) at line 1. Note that the beam search is performed on the original lattice which consists of  $L$  active labels per position. The beam search time is negligible compared to the total decoding time. At line 16, we update  $lb$  as follows. We enumerate the best active sequences backtracked by the nodes at position  $T - 1$ . If the current  $lb$  is less than the  $k$ -th active sequence score, we update the  $lb$  with the  $k$ -th active sequence score (we do not update  $lb$  if there are less than  $k$  active sequences). At line 19, we use the sequences returned from Viterbi A\* algorithm to update the  $lb$  in the same manner. To enable this update, we request the Viterbi A\* algorithm to return  $k'$ ,  $k' > k$ , sequences (line 10). A larger number of  $k'$  results in a higher chance to find the  $k$ -th active sequence,

which in turn offers a tighter (higher)  $lb$ , but it comes with the expense of additional time (the backward A\* process takes  $O(TL)$  time to return one more sequence). In experiments, we found the  $lb$  updates on line 1 and line 16 are essential for fast decoding. The updating of  $lb$  using Viterbi A\* sequences (line 19) can boost the decoding speed further. We experimented with different  $k'$  values ( $k' = nk$ , where  $n$  is an integer) and selected  $k' = 2k$  which results in the largest decoding speed boost.

---

#### Algorithm 4 $K$ -Best iterative Viterbi A\* algorithm

---

```

1:  $lb = k$ -th best (original lattice)
2: init lattice
3: for  $i = 0$ ;  $i++$  do
4:   if  $i \% 2 == 0$  then
5:      $y = forward()$ 
6:   else
7:      $y = backward()$ 
8:   end if
9:   if  $y$  consists of active labels only then
10:     $ys = k$ -best Viterbi A* (Algorithm 3)
11:    if  $ys$  consists of active sequences only then
12:      return  $ys$ 
13:    end if
14:   end if
15:   if  $lb < k$ -th best(lattice) then
16:      $lb = k$ -th best(lattice)
17:   end if
18:   if  $lb < k$ -th best( $ys$ ) then
19:      $lb = k$ -th best( $ys$ )
20:   end if
21:   expand lattice
22: end for

```

---

## 5 Experiments

We compare aforementioned 1-best and  $k$ -best sequential decoding algorithms using five datasets in this section.

### 5.1 Experimental setting

We apply 1-best and  $k$ -best sequential decoding algorithms to five NLP tagging tasks: Penn TreeBank (PTB) POS tagging, CoNLL2000 joint POS tagging and chunking, CoNLL 2003 joint POS tagging, chunking and named entity tagging, HPSG supertagging (Matsuzaki et al., 2007) and a search query named entity recognition (NER) dataset. We used

sections 02-21 of PTB for training and section 23 for testing in POS task. As in (Kaji et al., 2010), we combine the POS tags and chunk tags to form joint tags for CoNLL 2000 dataset, e.g., NN|B-NP. Similarly we combine the POS tags, chunk tags, and named entity tags to form joint tags for CoNLL 2003 dataset, e.g., PRP\$|I-NP|O. Note that by such tag joining, we are able to offer different tag decodings (for example, chunking and named entity tagging) simultaneously. This indeed is one of the effective approaches for joint tag decoding problems. The search query NER dataset is an in-house annotated dataset which assigns semantic labels, such as *product*, *business* tags to web search queries.

Table 2 shows the training and test sets size (sentence #), the average token length of test dataset and the label size for the five datasets. POS and supertag datasets assign tags to tokens while CoNLL 2000, CoNLL 2003 and search query datasets assign tags to phrases. We use the standard BIO encoding for CoNLL 2000, CoNLL 2003 and search query datasets.

Table 2: Training and test datasets size, average token length of test set and label size for five datasets.

	training #	test #	token length	label size
POS	39831	2415	23	45
CoNLL2000	8936	2012	23	319
CoNLL2003	14987	3684	12	443
Supertag	37806	2291	22	2602
search query	79569	6867	3	323

Due to the long CRF training time (days to weeks even for stochastic gradient descent training) for these large label size datasets, we choose the perceptron algorithm for training. The models are averaged over 10 iterations (Collins, 2002). The training time takes minutes to hours for all datasets. We note that the selection of training algorithm does not affect the decoding process: the decoding is identical for both CRF and perceptron training algorithms. We use the common features which are adopted in previous studies, for example (Sha and Periera, 2003). In particular, we use the unigrams of the current and its neighboring words, word bigrams, prefixes and suffixes of the current word, capitalization, all-number, punctuation, and tag bigrams for POS, CoNLL2000 and CoNLL 2003 datasets. For supertag dataset, we use the same features for the word inputs, and the unigrams and bigrams for gold POS inputs. For search query dataset, we use the same features plus gazetteer based features.

## 5.2 Results

We report the token accuracy for all datasets to facilitate comparison to previous work. They are 97.00, 94.70, 95.80, 90.60 and 88.60 for POS, CoNLL 2000, CoNLL 2003, supertag, and search query re-

spectively. We note that all decoding algorithms as listed in Section 3 and Section 4 are exact. That is, they produce exactly the same accuracy. The accuracy we get for the first four tasks is comparable to the state-of-the-art. We do not have a baseline to compare with for the last dataset as it is not publicly available<sup>7</sup>. Higher accuracy may be achieved if more task specific features are introduced on top of the standard features. As this paper is more concerned with the decoding speed, the feature engineering is beyond the scope of this paper.

Table 3 shows how many iterations in average are required for iterative Viterbi and iterative Viterbi A\* algorithms. Although the max iteration size is bounded to  $\lceil \log_2 L \rceil$  for each position (for example, 9 for CoNLL 2003 dataset), the total iteration number for the whole lattice may be greater than  $\lceil \log_2 L \rceil$  as different positions may not expand at the same time. Despite the large number of iterations used in iterative based algorithms (especially iterative Viterbi A\* algorithm), the algorithms are still very efficient (see below).

Table 3: Iteration numbers of iterative Viterbi and iterative Viterbi A\* algorithms for five datasets.

	POS	CoNLL2000	CoNLL2003	Supertag	search query
iter Viter	6.32	8.76	9.18	10.63	6.71
iter Viter A*	14.42	16.40	15.41	18.62	9.48

Table 4 and 5 show the decoding speed (sentences per second) of 1-best and 5-best decoding algorithms respectively. The proposed decoding algorithms and the largest decoding speeds across different decoding algorithms (other than beam) are highlighted in bold. We exclude the time for feature extraction in computing the speed. The beam search decoding is also shown as a baseline. We note that beam decoding is the only approximate decoding algorithm in this table. All other decoding algorithms produce exactly the same accuracy, which is usually much better than the accuracy of beam decoding.

For 1-best decoding, iterative Viterbi always outperforms other ones. A\* with a proper heuristic denoted as A\* (*best*), that is, the best A\* using naive heuristic or the values of  $m$  being 0, 3, 6 or 9 (see Section 4.1), can be the second best choice (except for the POS task), although the gap between iterative Viterbi and A\* is significant. For example, for CoNLL 2003 dataset, the former can decode 2239 sentences per second while the latter only decodes 225 sentences per second. The iterative process successfully boosts the decoding speed of iterative Viterbi compared to Viterbi, but it slows down the decoding speed of iterative A\* compared

<sup>7</sup>The lower accuracy is due to the dynamic nature of queries: many of test query tokens are unseen in the training set.

to A\*(best). This is because in the Viterbi case, the iterative process has a node pruning procedure, while it does not have such pruning in A\*(best) algorithm. Take CoNLL 2003 data as an example, the removal of the pruning slows down the 1-best iterative Viterbi decoding from 2239 to 604 sentences/second. Carpediem algorithm performs poorly in four out of five tasks. This can be explained as follows. The Carpediem implicitly assumes that the node scores are the dominant factors to determine the best sequence. However, this assumption does not hold as the edge scores play an important role.

For 5-best decoding,  $k$ -best Viterbi decoding is very slow. A\* with a proper heuristic is still slow. For example, it only reaches 11 sentences per second for CoNLL 2003 dataset. The classic Viterbi A\* can usually obtain a decent decoding speed, for example, 40 sentences per second for CoNLL 2003 dataset. The only exception is supertag dataset, on which the Viterbi A\* decodes 0.1 sentence per second while the A\* decodes 3. This indicates the scalability issue of Viterbi A\* algorithm for datasets with more than one thousand labels. The proposed iterative Viterbi A\* is clearly the winner. It speeds up the Viterbi A\* to factors of 4, 7, 360, and 3 for CoNLL 2000, CoNLL 2003, supertag and query search data respectively. The decoding speed of iterative Viterbi A\* can even be comparable to that of beam search.

Figure 4 shows  $k$ -best decoding algorithms decoding speed with respect to different  $k$  values for CoNLL 2003 data. The Viterbi A\* and iterative Viterbi A\* algorithms are significantly faster than the Viterbi and A\*(best) algorithms. Although the iterative Viterbi A\* significantly outperforms the Viterbi A\* for  $k < 30$ , the speed of the former converges to the latter when  $k$  becomes 90 or larger. This is expected as the  $k$ -best sequences span over the whole lattice: the earlier iteration in iterative Viterbi A\* algorithm cannot provide the  $k$ -best sequences using the degenerate lattice. The overhead of multiple iterations slows down the decoding speed compared to the Viterbi A\* algorithm.

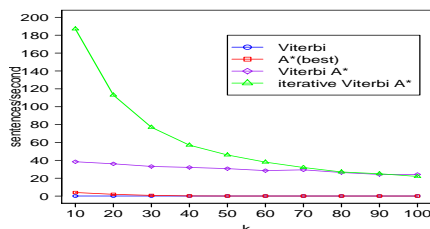


Figure 4: Decoding speed of  $k$ -best decoding algorithms for various  $k$  for CoNLL 2003 dataset.

## 6 Related work

The Viterbi algorithm is the only exact algorithm widely adopted in the NLP applications. Esposito and Radicioni (2009) proposed an algorithm which opens necessary nodes in a lattice in searching the best sequence. The staggered decoding (Kaji et al., 2010) forms the basis for our work on iterative based decoding algorithms. Apart from the exact decoding, approximate decoding algorithms such as beam search are also related to our work. Tsuruoka and Tsujii (2005) proposed easiest-first deterministic decoding. Siddiqi and Moore (2005) presented the parameter tying approach for fast inference in HMMs. A similar idea was applied to CRFs as well (Cohn, 2006; Jeong, 2009). We note that the exact algorithm always guarantees the optimality which cannot be attained in approximate algorithms.

In terms of  $k$ -best parsing, Huang and Chiang (2005) proposed an efficient algorithm which is similar to the  $k$ -best Viterbi A\* algorithm presented in this paper. Pauls and Klein (2009) proposed an algorithm which replaces the Viterbi forward pass with an A\* search. Their algorithm optimizes the Viterbi pass, while the proposed iterative Viterbi A\* algorithm optimizes both Viterbi and A\* passes.

This paper is also related to the coarse to fine PCFG parsing (Charniak et al., 2006) as the degenerate labels can be treated as coarse levels. However, the difference is that the coarse-to-fine parsing is an approximate decoding while ours is exact one. In terms of different coarse levels of heuristic used in A\* decoding, this paper is related to the work of hierarchical A\* framework (Raphael, 2001; Felzenszwalb et al., 2007). In terms of iterative process, this paper is close to (Burkett et al., 2011) as both exploit the search-and-expand approach.

## 7 Conclusions

We have presented and evaluated the A\* and iterative A\* algorithms for 1-best sequential decoding in this paper. In addition, we proposed A\* and iterative Viterbi A\* algorithm for  $k$ -best sequential decoding.  $K$ -best Iterative A\* algorithm can be several times or orders of magnitude faster than the state-of-the-art  $k$ -best decoding algorithm. It makes real-time large-scale tagging applications with thousands of labels feasible.

## Acknowledgments

We wish to thank Yusuke Miyao and Nobuhiro Kaji for providing us the HPSG Treebank data. We are grateful for the invaluable comments offered by the anonymous reviewers.

Table 4: Decoding speed (sentences per second) of 1-best decoding algorithms for five datasets.

	POS	CoNLL2000	CoNLL2003	supertag	query search
beam	7252	1381	1650	395	7571
Viterbi	2779	51	41	0.19	443
iterative Viterbi	<b>5833</b>	<b>972</b>	<b>2239</b>	<b>213</b>	<b>6805</b>
Carpediem	2638	14	20	0.15	243
<b>A* (best)</b>	802	131	225	8	880
<b>iterative A*</b>	1112	84	109	3	501

Table 5: Decoding speed (sentences per second) of 5-best decoding algorithms for five datasets.

	POS	CoNLL2000	CoNLL2003	supertag	query search
beam	2760	461	592	75	4354
Viterbi	19	0.41	0.25	0.12	3.83
<b>A* (best)</b>	205	4	11	3	92
Viterbi A*	<b>1266</b>	47	40	0.1	357
<b>iterative Viterbi A*</b>	788	<b>200</b>	<b>295</b>	<b>36</b>	<b>1025</b>

## References

- D. Burkett, D. Hall, and D. Klein. 2011. *Optimal graph search with iterated graph cuts*. Proceedings of AAAI.
- E. Charniak, M. Johnson, M. Elsnar, J. Austerweil, D. Ellis, I. Haxton, C. Hill, R. Shrivaths, J. Moore, M. Pozar, and T. Vu. 2006. *Multi-level coarse-to-fine PCFG parsing*. Proceedings of NAACL.
- T. Cohn. 2006. *Efficient inference in large conditional random fields*. Proceedings of ECML.
- M. Collins. 2002. *Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms*. Proceedings of EMNLP.
- R. Esposito and D. P. Radicioni. 2009. *Carpediem: Optimizing the Viterbi Algorithm and Applications to Supervised Sequential Learning*. Journal of Machine Learning Research.
- P. Felzenszwalb and D. McAllester. 2007. *The generalized A\* architecture*. Journal of Artificial Intelligence Research.
- P. E. Hart, N. J. Nilsson, and B. Raphael. 1968. *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*. IEEE Transactions on Systems Science and Cybernetics.
- L. Huang and D. Chiang. 2005. *Better k-best parsing*. Proceedings of the International Workshops on Parsing Technologies (IWPT).
- M. Jeong, C. Y. Lin, and G. G. Lee. 2009. *Efficient inference of CRFs for large-scale natural language data*. Proceedings of ACL-IJCNLP Short Papers.
- N. Kaji, Y. Fujiwara, N. Yoshinaga, and M. Kitsuregawa. 2010. *Efficient Staggered Decoding for Sequence Labeling*. Proceedings of ACL.
- D. Klein and C. Manning. 2003. *A\* parsing: Fast exact Viterbi parse selection*. Proceedings of ACL.
- J. Lafferty, A. McCallum, and F. Pereira. 2001. *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. Proceedings of ICML.
- T. Matsuzaki, Y. Miyao, and J. Tsujii. 2007. *Efficient HPSG parsing with supertagging and CFG-filtering*. Proceedings of IJCAI.
- A. Pauls and D. Klein. 2009. *K-Best A\* Parsing*. Proceedings of ACL.
- L. R. Rabiner. 1989. *A tutorial on hidden Markov models and selected applications in speech recognition*. Proceedings of The IEEE.
- C. Raphael. 2001. *Coarse-to-fine dynamic programming*. IEEE Transactions on Pattern Analysis and Machine Intelligence.
- S. Russell and P. Norvig. 1995. *Artificial Intelligence: A Modern Approach*.
- F. Sha and F. Pereira. 2003. *Shallow parsing with conditional random fields*. Proceedings of HLT-NAACL.
- S. M. Siddiqi and A. Moore. 2005. *Fast inference and learning in large-state-space HMMs*. Proceedings of ICML.
- Y. Tsuruoka and J. Tsujii. 2005. *Bidirectional inference with the easiest-first strategy for tagging sequence data*. Proceedings of HLT/EMNLP.
- A. J. Viterbi. 1967. *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm*. IEEE Transactions on Information Theory.

# Bootstrapping via Graph Propagation

Max Whitney and Anoop Sarkar\*

Simon Fraser University, School of Computing Science  
Burnaby, BC V5A 1S6, Canada

{mwhitney, anoop}@sfu.ca

## Abstract

Bootstrapping a classifier from a small set of seed rules can be viewed as the propagation of labels between examples via features shared between them. This paper introduces a novel variant of the Yarowsky algorithm based on this view. It is a bootstrapping learning method which uses a graph propagation algorithm with a well defined objective function. The experimental results show that our proposed bootstrapping algorithm achieves state of the art performance or better on several different natural language data sets.

## 1 Introduction

In this paper, we are concerned with a case of semi-supervised learning that is close to unsupervised learning, in that the labelled and unlabelled data points are from the same domain and only a small set of seed rules is used to derive the labelled points. We refer to this setting as *bootstrapping*. In contrast, typical semi-supervised learning deals with a large number of labelled points, and a domain adaptation task with unlabelled points from the new domain.

The two dominant discriminative learning methods for bootstrapping are self-training (Scudder, 1965) and co-training (Blum and Mitchell, 1998). In this paper we focus on a self-training style bootstrapping algorithm, the Yarowsky algorithm (Yarowsky, 1995). Variants of this algorithm have been formalized as optimizing an objective function in previous work by Abney (2004) and Haffari and Sarkar (2007), but it is not clear that any perform as well as the Yarowsky algorithm itself.

We take advantage of this formalization and introduce a novel algorithm called Yarowsky-prop which builds on the algorithms of Yarowsky (1995) and Subramanya et al. (2010). It is theoretically

\*This research was partially supported by an NSERC, Canada (RGPIN: 264905) grant. We would like to thank Gholamreza Haffari and the anonymous reviewers for their comments. We particularly thank Michael Collins, Jason Eisner, and Damianos Karakos for the data we used in our experiments.

$x$	denotes an example
$f, g$	denote features
$i, k$	denote labels
$X$	set of training examples
$F_x$	set of features for example $x$
$Y$	current labelling of $X$
$Y_x$	current label for example $x$
$\perp$	value of $Y_x$ for unlabelled examples
$L$	number of labels (not including $\perp$ )
$\Lambda$	set of currently labelled examples
$V$	set of currently unlabelled examples
$X_f$	set of examples with feature $f$
$\Lambda_f$	set of currently labelled examples with $f$
$V_f$	set of currently unlabelled examples with $f$
$\Lambda_j$	set of examples currently labelled with $j$
$\Lambda_{fj}$	set of examples with $f$ currently labelled with $j$

Table 1: Notation of Abney (2004).

well-understood as minimizing an objective function at each iteration, and it obtains state of the art performance on several different NLP data sets. To our knowledge, this is the first theoretically motivated self-training bootstrapping algorithm which performs as well as the Yarowsky algorithm.

## 2 Bootstrapping

Abney (2004) defines useful notation for semi-supervised learning, shown in table 1. Note that  $\Lambda$ ,  $V$ , etc. are relative to the current labelling  $Y$ . We additionally define  $F$  to be the set of all features, and use  $U$  to denote the uniform distribution. In the bootstrapping setting the learner is given an initial partial labelling  $Y^{(0)}$  where only a few examples are labelled (i.e.  $Y_x^{(0)} = \perp$  for most  $x$ ).

Abney (2004) defines three probability distributions in his analysis of bootstrapping:  $\theta_{fj}$  is the parameter for feature  $f$  with label  $j$ , taken to be normalized so that  $\theta_f$  is a distribution over labels.  $\phi_x$  is the labelling distribution representing the current  $Y$ ; it is a point distribution for labelled examples and uniform for unlabelled examples.  $\pi_x$  is the prediction distribution over labels for example  $x$ .

The approach of Haghighi and Klein (2006b) and Haghighi and Klein (2006a) also uses a small set of



---

Algorithm 1: The basic Yarowsky algorithm.

---

**Require:** training data  $X$  and a seed DL  $\theta^{(0)}$

- 1: apply  $\theta^{(0)}$  to  $X$  produce a labelling  $Y^{(0)}$
  - 2: **for** iteration  $t$  to maximum or convergence **do**
  - 3:   train a new DL  $\theta$  on  $Y^{(t)}$
  - 4:   apply  $\theta$  to  $X$ , to produce  $Y^{(t+1)}$
  - 5: **end for**
- 

seed rules but uses them to inject features into a joint model  $p(x, j)$  which they train using expectation-maximization for Markov random fields. We focus on discriminative training which does not require complex partition functions for normalization. Blum and Chawla (2001) introduce an early use of transductive learning using graph propagation. X. Zhu and Z. Ghahramani and J. Lafferty (2003)’s method of graph propagation is predominantly transductive, and the non-transductive version is closely related to Abney (2004) c.f. Haffari and Sarkar (2007).<sup>1</sup>

### 3 Existing algorithms

#### 3.1 Yarowsky

A decision list (DL) is a (ordered) list of feature-label pairs (rules) which is produced by assigning a score to each rule and sorting on this score. It chooses a label for an example from the first rule whose feature is a feature of the example. For a DL the prediction distribution is defined by  $\pi_x(j) \propto \max_{f \in F_x} \theta_{fj}$ . The basic Yarowsky algorithm is shown in algorithm 1. Note that at any point some training examples may be left unlabelled by  $Y^{(t)}$ .

We use Collins and Singer (1999) for our exact specification of Yarowsky.<sup>2</sup> It uses DL rule scores

$$\theta_{fj} \propto \frac{|\Lambda_{fj}| + \epsilon}{|\Lambda_f| + L\epsilon} \quad (1)$$

where  $\epsilon$  is a smoothing constant. When constructing a DL it keeps only the rules with (pre-normalized) score over a threshold  $\zeta$ . In our implementation we add the seed rules to each subsequent DL.<sup>3</sup>

---

<sup>1</sup>Large-scale information extraction, e.g. (Hearst, 1992), Snowball (Agichtein and Gravano, 2000), AutoSlog (Riloff and Shepherd, 1997), and Junto (Talukdar, 2010) among others, also have similarities to our approach. We focus on the formal analysis of the Yarowsky algorithm by Abney (2004).

<sup>2</sup>It is similar to that of Yarowsky (1995) but is better specified and omits word sense disambiguation optimizations. The general algorithm in Yarowsky (1995) is self-training with any kind of underlying supervised classifier, but we follow the convention of using Yarowsky to refer to the DL algorithm.

<sup>3</sup>This is not clearly specified in Collins and Singer (1999),

#### 3.2 Yarowsky-cautious

Collins and Singer (1999) also introduce a variant algorithm Yarowsky-cautious. Here the DL training step keeps only the top  $n$  rules  $(f, j)$  over the threshold for each label  $j$ , ordered by  $|\Lambda_f|$ . Additionally the threshold  $\zeta$  is checked against  $|\Lambda_{fj}|/|\Lambda_f|$  instead of the smoothed score.  $n$  begins at  $n_0$  and is incremented by  $\Delta n$  at each iteration. We add the seed DL to the new DL after applying the cautious pruning. Cautiousness limits not only the size of the DL but also the number of labelled examples, prioritizing decisions which are believed to be of high accuracy.

At the final iteration Yarowsky-cautious uses the current labelling to train a DL without a threshold or cautiousness, and this DL is used for testing. We call this the retraining step.<sup>4</sup>

#### 3.3 DL-CoTrain

Collins and Singer (1999) also introduce the co-training algorithm DL-CoTrain. This algorithm alternates between two DLs using disjoint views of the features in the data. At each step it trains a DL and then produces a new labelling for the other DL. Each DL uses thresholding and cautiousness as we describe for Yarowsky-cautious. At the end the DLs are combined, the result is used to label the data, and a retraining step is done from this single labelling.

#### 3.4 Y-1/DL-1-VS

One of the variant algorithms of Abney (2004) is Y-1/DL-1-VS (referred to by Haffari and Sarkar (2007) as simply DL-1). Besides various changes in the specifics of how the labelling is produced, this algorithm has two differences versus Yarowsky. Firstly, the smoothing constant  $\epsilon$  in (1) is replaced by  $1/|V_f|$ . Secondly,  $\pi$  is redefined as  $\pi_x(j) = \frac{1}{|F_x|} \sum_{f \in F_x} \theta_{fj}$ , which we refer to as the sum definition of  $\pi$ . This definition does not match a literal DL but is easier to analyze.

We are not concerned here with the details of Y-1/DL-1-VS, but we note that Haffari and Sarkar

---

but is used for DL-CoTrain in the same paper.

<sup>4</sup>The details of Yarowsky-cautious are not clearly specified in Collins and Singer (1999). Based on similar parts of DL-CoTrain we assume that the top  $n$  selection is per label rather than in total, that the thresholding value is unsmoothed, and that there is a retraining step. We also assume their notation  $Count'(x)$  to be equivalent to  $|\Lambda_f|$ .

(2007) provide an objective function for this algorithm using a generalized definition of cross-entropy in terms of Bregman distance, which motivates our objective in section 4. The Bregman distance between two discrete probability distributions  $p$  and  $q$  is defined as  $B_\psi(p, q) = \sum_i [\psi(p_i) - \psi(q_i) - \psi'(q_i)(p_i - q_i)]$ . As a specific case we have  $B_{t^2}(p, q) = \sum_i (p_i - q_i)^2 = \|p - q\|^2$ . Then Bregman distance-based entropy is  $H_{t^2}(p) = -\sum_i p_i^2$ , KL-Divergence is  $B_{t^2}$ , and cross-entropy follows the standard definition in terms of  $H_{t^2}$  and  $B_{t^2}$ . The objective minimized by Y-1/DL-1-VS is:

$$\sum_{\substack{x \in X \\ f \in F_x}} H_{t^2}(\phi_x || \theta_f) = \sum_{\substack{x \in X \\ f \in F_x}} \left[ B_{t^2}(\phi_x || \theta_f) - \sum_y \phi_x^2 \right]. \quad (2)$$

### 3.5 Yarowsky-sum

As a baseline for the sum definition of  $\pi$ , we introduce the Yarowsky-sum algorithm. It is the same as Yarowsky except that we use the sum definition when labelling: for example  $x$  we choose the label  $j$  with the highest (sum)  $\pi_x(j)$ , but set  $Y_x = \perp$  if the sum is zero. Note that this is a linear model similar to a conditional random field (CRF) (Lafferty et al., 2001) for unstructured multiclass problems.

### 3.6 Bipartite graph algorithms

Haffari and Sarkar (2007) suggest a bipartite graph framework for semi-supervised learning based on their analysis of Y-1/DL-1-VS and objective (2). The graph has vertices  $X \cup F$  and edges  $\{(x, f) : x \in X, f \in F_x\}$ , as in the graph shown in figure 1(a). Each vertex represents a distribution over labels, and in this view Yarowsky can be seen as alternately updating the example distributions based on the feature distributions and visa versa.

Based on this they give algorithm 2, which we call HS-bipartite. It is parametrized by two functions which are called features-to-example and examples-to-feature here. Each can be one of two choices: **average**( $S$ ) is the normalized average of the distributions of  $S$ , while **majority**( $S$ ) is a uniform distribution if all labels are supported by equal numbers of distributions of  $S$ , and otherwise a point distribution with mass on the best supported label. The **average-majority** form is similar

---

Algorithm 2: HS-bipartite.

---

```

1: apply  $\theta^{(0)}$  to  $X$  produce a labelling  $Y^{(0)}$ 
2: for iteration  $t$  to maximum or convergence do
3:   for  $f \in F$  do
4:     let  $p = \text{examples-to-feature}(\{\phi_x : x \in X_f\})$ 
5:     if  $p \neq U$  then let  $\theta_f = p$ 
6:   end for
7:   for  $x \in X$  do
8:     let  $p = \text{features-to-example}(\{\theta_f : f \in F_x\})$ 
9:     if  $p \neq U$  then let  $\phi_x = p$ 
10:  end for
11: end for

```

---

to Y-1/DL-1-VS, and the **majority-majority** form minimizes a different objective similar to (2).

In our implementation we label training data (for the convergence check) with the  $\phi$  distributions from the graph. We label test data by constructing new  $\phi_x = \text{examples-to-feature}(F_x)$  for the unseen  $x$ .

### 3.7 Semi-supervised learning algorithm of Subramanya et al. (2010)

Subramanya et al. (2010) give a semi-supervised algorithm for part of speech tagging. Unlike the algorithms described above, it is for domain adaptation with large amounts of labelled data rather than bootstrapping with a small number of seeds.

This algorithm is structurally similar to Yarowsky in that it begins from an initial partial labelling and repeatedly trains a classifier on the labelling and then relabels the data. It uses a CRF (Lafferty et al., 2001) as the underlying supervised learner. It differs significantly from Yarowsky in two other ways: First, instead of only training a CRF it also uses a step of graph propagation between distributions over the  $n$ -grams in the data. Second, it does the propagation on distributions over  $n$ -gram types rather than over  $n$ -gram tokens (instances in the data).

They argue that using propagation over types allows the algorithm to enforce constraints and find similarities that self-training cannot. We are not concerned here with the details of this algorithm, but it motivates our work firstly in providing the graph propagation which we will describe in more detail in section 4, and secondly in providing an algorithmic structure that we use for our algorithm in section 5.

### 3.8 Collins and Singer (1999)'s EM

We implemented the EM algorithm of Collins and Singer (1999) as a baseline for the other algorithms.

Method	$\mathcal{V}$	$\mathcal{N}(u)$	$q_u$
$\phi$ - $\theta$	$X \cup F$	$\mathcal{N}_x = F_x, \mathcal{N}_f = X_f$	$q_x = \phi_x, q_f = \theta_f$
$\pi$ - $\theta$	$X \cup F$	$\mathcal{N}_x = F_x, \mathcal{N}_f = X_f$	$q_x = \pi_x, q_f = \theta_f$
$\theta$ -only	$F$	$\mathcal{N}_f = \bigcup_{x \in X_f} F_x \setminus f$	$q_f = \theta_f$
$\theta^T$ -only	$F$	$\mathcal{N}_f = \bigcup_{x \in X_f} F_x \setminus f$	$q_f = \theta_f^T$

Table 2: Graph structures for propagation.

They do not specify tuning details, but to get comparable accuracy we found it was necessary to do smoothing and to include weights  $\lambda_1$  and  $\lambda_2$  on the expected counts of seed-labelled and initially unlabelled examples respectively (Nigam et al., 2000).

## 4 Graph propagation

The graph propagation of Subramanya et al. (2010) is a method for smoothing distributions attached to vertices of a graph. Here we present it with an alternate notation using Bregman distances as described in section 3.4.<sup>5</sup> The objective is

$$\mu \sum_{\substack{u \in \mathcal{V} \\ v \in \mathcal{N}(i)}} w_{uv} B_{t^2}(q_u, q_v) + \nu \sum_{u \in \mathcal{V}} B_{t^2}(q_u, U) \quad (3)$$

where  $\mathcal{V}$  is a set of vertices,  $\mathcal{N}(v)$  is the neighbourhood of vertex  $v$ , and  $q_v$  is an initial distribution for each vertex  $v$  to be smoothed. They give an iterative update to minimize (3). Note that (3) is independent of their specific graph structure, distributions, and semi-supervised learning algorithm.

We propose four methods for using this propagation with Yarowsky. These methods all use constant edge weights ( $w_{uv} = 1$ ). The distributions and graph structures are shown in table 2. Figure 1 shows example graphs for  $\phi$ - $\theta$  and  $\theta$ -only.  $\pi$ - $\theta$  and  $\theta^T$ -only are similar, and are described below.

The graph structure of  $\phi$ - $\theta$  is the bipartite graph of Haffari and Sarkar (2007). In fact,  $\phi$ - $\theta$  the propagation objective (3) and Haffari and Sarkar (2007)’s Y-1/DL-1-VS objective (2) are identical up to constant coefficients and an extra constant term.<sup>6</sup>  $\phi$ - $\theta$

<sup>5</sup>We omit the option to hold some of the distributions at fixed values, which would add an extra term to the objective.

<sup>6</sup>The differences are specifically: First, (3) adds the constant coefficients  $\mu$  and  $\nu$ . Second, (3) sums over each edge twice (once in each direction), while (2) sums over each only once. Since  $w_{uv} = w_{vu}$  and  $B_{t^2}(q_u, q_v) = B_{t^2}(q_v, q_u)$ , this can be folded into the constant  $\mu$ . Third, after expanding (2) there is a term  $|F_x|$  inside the sum for  $H_{t^2}(\phi_x)$  which is not present in (3). This does not effect the direction of minimization. Fourth,  $B_{t^2}(q_u, U)$  in (3) expands to  $H_{t^2}(q_u)$  plus a constant, adding an extra constant term to the total.

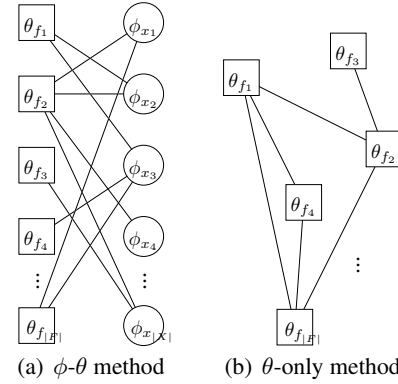


Figure 1: Example graphs for  $\phi$ - $\theta$  and  $\theta$ -only propagation.

therefore gives us a direct way to optimize (2).

The other three methods do not correspond to the objective of Haffari and Sarkar (2007). The  $\pi$ - $\theta$  method is like  $\phi$ - $\theta$  except for using  $\pi$  as the distribution for example vertices.

The bipartite graph of the first two methods differs from the structure used by Subramanya et al. (2010) in that it does propagation between two different kinds of distributions instead of only one kind. We also adopt a more comparable approach with a graph over only features. Here we define adjacency by co-occurrence in the same example. The  $\theta$ -only method uses this graph and  $\theta$  as the distribution.

Finally, we noted in section 3.7 that the algorithm of Subramanya et al. (2010) does one additional step in converting from token level distributions to type level distributions. The  $\theta^T$ -only method therefore uses the feature-only graph but for the distribution uses a type level version of  $\theta$  defined by  $\theta_{fj}^T = \frac{1}{|X_f|} \sum_{x \in X_f} \pi_x(j)$ .

## 5 Novel Yarowsky-prop algorithm

We call our graph propagation based algorithm Yarowsky-prop. It is shown with  $\theta^T$ -only propagation in algorithm 3. It is based on the Yarowsky algorithm, with the following changes: an added step to calculate  $\theta^T$  (line 4), an added step to calculate  $\theta^P$  (line 5), the use of  $\theta^P$  rather than the DL to update the labelling (line 6), and the use of the sum definition of  $\pi$ . Line 7 does DL training as we describe in sections 3.1 and 3.2. Propagation is done with the iterative update of Subramanya et al. (2010).

This algorithm is adapted to the other propagation methods described in section 4 by changing the type of propagation on line 5. In  $\theta$ -only, propagation is

Algorithm 3: Yarowsky-prop.

---

```

1: let  $\theta_{fj}$  be the scores of the seed rules // crf_train
2: for iteration  $t$  to maximum or convergence do
3:   let  $\pi_x(j) = \frac{1}{|F_x|} \sum_{f \in F_x} \theta_{fj}$  // post_decode
4:   let  $\theta_{fj}^T = \frac{\sum_{x \in X_f} \pi_x(j)}{|X_f|}$  // token_to_type
5:   propagate  $\theta^T$  to get  $\theta^P$  // graph_propagate
6:   label the data with  $\theta^P$  // viterbi_decode
7:   train a new DL  $\theta_{fj}$  // crf_train
8: end for

```

---

done on  $\theta$ , using the graph of figure 1(b). In  $\phi$ - $\theta$  and  $\pi$ - $\theta$  propagation is done on the respective bipartite graph (figure 1(a) or the equivalent with  $\pi$ ). Line 4 is skipped for these methods, and  $\phi$  is as defined in section 2. For the bipartite graph methods  $\phi$ - $\theta$  and  $\pi$ - $\theta$  only the propagated  $\theta$  values on the feature nodes are used for  $\theta^P$  (the distributions on the example nodes are ignored after the propagation itself).

The algorithm uses  $\theta_{fj}$  values rather than an explicit DL for labelling. The (pre-normalized) score for any  $(f, j)$  not in the DL is taken to be zero. Besides using the sum definition of  $\pi$  when calculating  $\theta^T$ , we also use a sum in labelling. When labelling an example  $x$  (at line 6 and also on testing data) we use  $\arg \max_j \sum_{f \in F_x: \theta_f^P \neq U} \theta_{fj}^P$ , but set  $Y_x = \perp$  if the sum is zero. Ignoring uniform  $\theta_f^P$  values is intended to provide an equivalent to the DL behaviour of using evidence only from rules that are in the list.

We include the cautiousness of Yarowsky-cautious (section 3.2) in the DL training on line 7. At the labelling step on line 6 we label only examples which the pre-propagated  $\theta$  would also assign a label (using the same rules described above for  $\theta^P$ ). This choice is intended to provide an equivalent to the Yarowsky-cautious behaviour of limiting the number of labelled examples; most  $\theta_f^P$  are non-uniform, so without it most examples become labelled early.

We observe further similarity between the Yarowsky algorithm and the general approach of Subramanya et al. (2010) by comparing algorithm 3 here with their algorithm 1. The comments in algorithm 3 give the corresponding parts of their algorithm. Note that each line has a similar purpose.

## 6 Evaluation

### 6.1 Tasks and data

For evaluation we use the tasks of Collins and Singer (1999) and Eisner and Karakos (2005), with data

Rank	Score	Feature	Label
<b>1</b>	0.999900	New-York	loc.
<b>2</b>	0.999900	California	loc.
<b>3</b>	0.999900	U.S.	loc.
<b>4</b>	0.999900	Microsoft	org.
<b>5</b>	0.999900	I.B.M.	org.
<b>6</b>	0.999900	Incorporated	org.
<b>7</b>	0.999900	Mr.	per.
8	0.999976	U.S.	loc.
9	0.999957	New-York-Stock-Exchange	loc.
10	0.999952	California	loc.
11	0.999947	New-York	loc.
12	0.999946	<i>court-in</i>	loc.
13	0.975154	<i>Company-of</i>	loc.
		$\vdots$	

Figure 2: A DL from iteration 5 of Yarowsky on the named entity task. Scores are pre-normalized values from the expression on the left side of (1), not  $\theta_{fj}$  values. Context features are indicated by *italics*; all others are spelling features. Specific feature types are omitted. Seed rules are indicated by **bold** ranks.

kindly provided by the respective authors.

The task of Collins and Singer (1999) is named entity classification on data from New York Times text.<sup>7</sup> The data set was pre-processed by a statistical parser (Collins, 1997) and all noun phrases that are potential named entities were extracted from the parse tree. Each noun phrase is to be labelled as a person, organization, or location. The parse tree provides the surrounding context as *context features* such as the words in prepositional phrase and relative clause modifiers, etc., and the actual words in the noun phrase provide the *spelling features*. The test data additionally contains some noise examples which are not in the three named entity categories. We use the seed rules the authors provide, which are the first seven items in figure 2. For DL-CoTrain, we use their two views: one view is the spelling features, and the other is the context features. Figure 2 shows a DL from Yarowsky training on this task.

The tasks of Eisner and Karakos (2005) are word sense disambiguation on several English words which have two senses corresponding to two different words in French. Data was extracted from the Canadian Hansards, using the English side to produce training and test data and the French side to produce the gold labelling. Features are the original and lemmatized words immediately adja-

<sup>7</sup>We removed weekday and month examples from the test set as they describe. They note 88962 examples in their training set, but the file has 89305. We did not find any filtering criteria that produced the expected size, and therefore used all examples.

cent to the word to be disambiguated, and original and lemmatized context words in the same sentence. Their seeds are pairs of adjacent word features, with one feature for each label (sense). We use the ‘drug’, ‘land’, and ‘sentence’ tasks, and the seed rules from their best seed selection: ‘alcohol’/‘medical’, ‘acres’/‘court’, and ‘reads’/‘served’ respectively (they do not provide seeds for their other three tasks). For DL-CoTrain we use adjacent words for one view and context words for the other.

## 6.2 Experimental set up

Where applicable we use smoothing  $\epsilon = 0.1$ , a threshold  $\zeta = 0.95$ , and cautiousness parameters  $n_0 = \Delta n = 5$  as in Collins and Singer (1999) and propagation parameters  $\mu = 0.6, \nu = 0.01$  as in Subramanya et al. (2010). Initial experiments with different propagation parameters suggested that as long as  $\nu$  was set at this value changing  $\mu$  had relatively little effect on the accuracy. We did not find any propagation parameter settings that outperformed this choice. For the Yarowsky-prop algorithms we perform a single iteration of the propagation update for each iteration of the algorithm.

For EM we use weights  $\lambda_1 = 0.98$ , and  $\lambda_2 = 0.02$  (see section 3.8), which were found in initial experiments to be the best values, and results are averaged over 10 random initializations.

The named entity test set contains some examples that are neither person, organization, nor location. Collins and Singer (1999) define noise accuracy as accuracy that includes such instances, and clean accuracy as accuracy calculated across only the examples which are one of the known labels. We report only clean accuracy in this paper; noise accuracy tracks clean accuracy but is a little lower. There is no difference on the word sense data sets. We also report (clean) non-seeded accuracy, which we define to be clean accuracy over only examples which are not assigned a label by the seed rules. This is intended to evaluate what the algorithm has learned, rather than what it can achieve by using the input information directly (Daume, 2011).

We test Yarowsky, Yarowsky-cautious, Yarowsky-sum, DL-CoTrain, HS-bipartite in all four forms, and Yarowsky-prop cautious and non-cautious and in all four forms. For each algorithm except EM we perform a final retraining step

Gold	Spelling features	Context features
loc.	Waukegan	<i>maker, LEFT</i>
loc.	Mexico, president, of	<i>president-of, RIGHT</i>
loc.	La-Jolla, La Jolla	<i>company, LEFT</i>

Figure 3: Named entity test set examples where Yarowsky-prop  $\theta$ -only is correct and no other tested algorithms are correct. The specific feature types are omitted.

as described for Yarowsky-cautious (section 3.2). Our programs and experiment scripts have been made available.<sup>8</sup>

## 6.3 Accuracy

Table 3 shows the final test set accuracies for the all the algorithms. The seed DL accuracy is also included for reference.

The best performing form of our novel algorithm is Yarowsky-prop-cautious  $\theta$ -only. It numerically outperforms DL-CoTrain on the named entity task, is not (statistically) significantly worse on the drug and land tasks, and is significantly better on the sentence task. It also numerically outperforms Yarowsky-cautious on the named entity task and is significantly better on the drug task. Is significantly worse on the land task, where most algorithms converge at labelling all examples with the first sense. It is significantly worse on the sentence task, although it is the second best performing algorithm and several percent above DL-CoTrain on that task.

Figure 3 shows (all) three examples from the named entity test set where Yarowsky-prop-cautious  $\theta$ -only is correct but none of the other Yarowsky variants are. Note that it succeeds despite misleading features; “maker” and “company” might be taken to indicate a company and “president-of” an organization, but all three examples are locations.

Yarowsky-prop-cautious  $\phi$ - $\theta$  and  $\pi$ - $\theta$  also perform respectably, although not as well. Yarowsky-prop-cautious  $\theta^T$ -only and the non-cautious versions are significantly worse. Although  $\theta^T$ -only was intended to incorporate Subramanya et al. (2010)’s idea of type level distributions, it in fact performs worse than  $\theta$ -only. We believe that Collins and Singer (1999)’s definition (1) of  $\theta$  incorporates sufficient type level information that the creation of a separate distribution is unnecessary in this case.

Figure 4 shows the test set non-seeded accuracies as a function of the iteration for many of the algo-

<sup>8</sup>The software is included with the paper submission and will be maintained at <https://github.com/sfu-naifang/yarowsky>.

Algorithm	Task							
	named entity		drug		land		sentence	
EM	81.05	78.64	55.96	54.85	32.86	31.07	67.88	65.42
	$\pm 0.31$	$\pm 0.34$	$\pm 0.41$	$\pm 0.43$	$\pm 0.00$	$\pm 0.00$	$\pm 3.35$	$\pm 3.57$
Seed DL	<i>11.29</i>	<i>0.00</i>	<i>5.18</i>	<i>0.00</i>	<i>2.89</i>	<i>0.00</i>	<i>7.18</i>	<i>0.00</i>
DL-CoTrain (cautious)	91.56	90.49	<b>59.59</b>	<b>58.17</b>	78.36	77.72	68.16	65.69
Yarowsky	<i>81.19</i>	<i>78.79</i>	55.70	54.02	<i>79.03</i>	<i>78.41</i>	<i>62.91</i>	<i>60.04</i>
Yarowsky-cautious	91.11	89.97	54.40	52.63	<b>79.10</b>	<b>78.48</b>	<b>78.64</b>	<b>76.99</b>
Yarowsky-cautious sum	91.56	90.49	54.40	52.63	78.36	77.72	<b>78.64</b>	<b>76.99</b>
HS-bipartite avg-avg	<i>45.84</i>	<i>45.89</i>	<i>52.33</i>	<i>50.42</i>	78.36	77.72	<i>54.56</i>	<i>51.05</i>
HS-bipartite avg-maj	<i>81.98</i>	<i>79.69</i>	<i>52.07</i>	<i>50.14</i>	78.36	77.72	<i>55.15</i>	<i>51.67</i>
HS-bipartite maj-avg	<i>73.55</i>	<i>70.18</i>	<i>52.07</i>	<i>50.14</i>	78.36	77.72	<i>55.15</i>	<i>51.67</i>
HS-bipartite maj-maj	<i>73.66</i>	<i>70.31</i>	<i>52.07</i>	<i>50.14</i>	78.36	77.72	<i>55.15</i>	<i>51.67</i>
Yarowsky-prop $\phi$ - $\theta$	<i>80.39</i>	<i>77.89</i>	<i>53.63</i>	<i>51.80</i>	78.36	77.72	<i>55.34</i>	<i>51.88</i>
Yarowsky-prop $\pi$ - $\theta$	<i>78.34</i>	<i>75.58</i>	<i>54.15</i>	<i>52.35</i>	78.36	77.72	<i>54.56</i>	<i>51.05</i>
Yarowsky-prop $\theta$ -only	<i>78.56</i>	<i>75.84</i>	<i>54.66</i>	<i>52.91</i>	78.36	77.72	<i>54.56</i>	<i>51.05</i>
Yarowsky-prop $\theta^T$ -only	<i>77.88</i>	<i>75.06</i>	<i>52.07</i>	<i>50.14</i>	78.36	77.72	<i>54.56</i>	<i>51.05</i>
Yarowsky-prop-cautious $\phi$ - $\theta$	90.19	88.95	56.99	55.40	78.36	77.72	<i>74.17</i>	<i>72.18</i>
Yarowsky-prop-cautious $\pi$ - $\theta$	<i>89.40</i>	<i>88.05</i>	58.55	57.06	78.36	77.72	70.10	67.78
Yarowsky-prop-cautious $\theta$ -only	<b>92.47</b>	<b>91.52</b>	58.55	57.06	78.36	77.72	<i>75.15</i>	<i>73.22</i>
Yarowsky-prop-cautious $\theta^T$ -only	<i>78.45</i>	<i>75.71</i>	58.29	56.79	78.36	77.72	<i>54.56</i>	<i>51.05</i>
Num. train/test examples	89305 / 962		134 / 386		1604 / 1488		303 / 515	

Table 3: Test set percent accuracy and non-seeded test set percent accuracy (respectively) for the algorithms on all tasks. **Bold** items are a maximum in their column. *Italic* items have a statistically significant difference versus DL-CoTrain ( $p < 0.05$  with a McNemar test). For EM,  $\pm$  indicates one standard deviation but statistical significance was not measured.

rithms on the named entity task. The Yarowsky-prop non-cautious algorithms quickly converge to the final accuracy and are not shown. While the other algorithms (figure 4(a)) make a large accuracy improvement in the final retraining step, the Yarowsky-prop (figure 4(b)) algorithms reach comparable accuracies earlier and gain much less from retraining.

We did not implement Collins and Singer (1999)’s CoBoost; however, in their results it performs comparably to DL-CoTrain and Yarowsky-cautious. As with DL-CoTrain, CoBoost requires two views.

#### 6.4 Cautiousness

Cautiousness appears to be important in the performance of the algorithms we tested. In table 3, only the cautious algorithms are able to reach the 90% accuracy range.

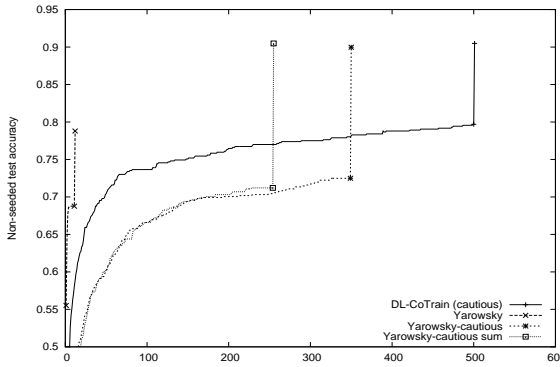
To evaluate the effects of cautiousness we examine the Yarowsky-prop  $\theta$ -only algorithm on the named entity task in more detail. This algorithm has two classifiers which are trained in conjunction: the DL and the propagated  $\theta^P$ . Figure 5 shows the training set coverage (of the labelling on line 6 of algorithm 3) and the test set accuracy of both classifiers, for the cautious and non-cautious versions.

The non-cautious version immediately learns a DL over all feature-label pairs, and therefore has full

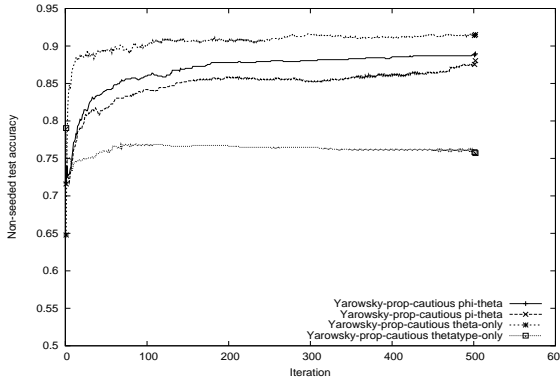
coverage after the first iteration. The DL and  $\theta^P$  converge to similar accuracies within a few more iterations, and the retraining step increases accuracy by less than one percent. On the other hand, the cautious version gradually increases the coverage over the iterations. The DL accuracy follows the coverage closely (similar to the behaviour of Yarowsky-cautious, not shown here), while the propagated classifier accuracy jumps quickly to near 90% and then increases only gradually.

Although the DL prior to retraining achieves a roughly similar accuracy in both versions, only the cautious version is able to reach the 90% accuracy range in the propagated classifier and retraining. Presumably the non-cautious version makes an early mistake, reaching a local minimum which it cannot escape. The cautious version avoids this by making only safe rule selection and labelling choices.

Figure 5(b) also helps to clarify the difference in retraining that we noted in section 6.3. Like the non-propagated DL algorithms, the DL component of Yarowsky-prop has much lower accuracy than the propagated classifier prior to the retraining step. But after retraining, the DL and  $\theta^P$  reach very similar accuracies.



(a) Collins & Singer algorithms (plus sum form)



(b) Yarowsky propagation cautious

Figure 4: Non-seeded test accuracy versus iteration for various algorithms on named entity. The results for the Yarowsky-prop algorithms are for the propagated classifier  $\theta^P$ , except for the final DL retraining iteration.

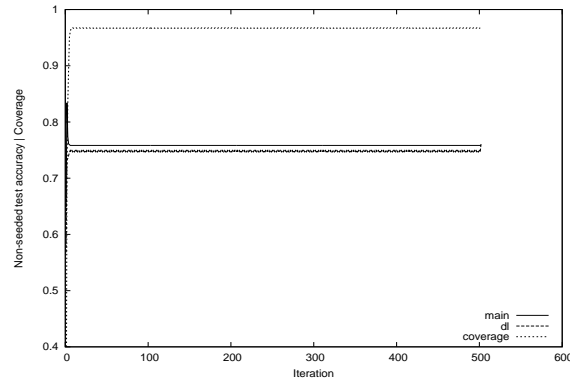
## 6.5 Objective function

The propagation method  $\phi$ - $\theta$  was motivated by optimizing the equivalent objectives (2) and (3) at each iteration. Figure 6 shows the graph propagation objective (3) along with accuracy for Yarowsky-prop  $\phi$ - $\theta$  without cautiousness. The objective value decreases as expected, and converges along with accuracy. Conversely, the cautious version (not shown here) does not clearly minimize the objective, since cautiousness limits the effect of the propagation.

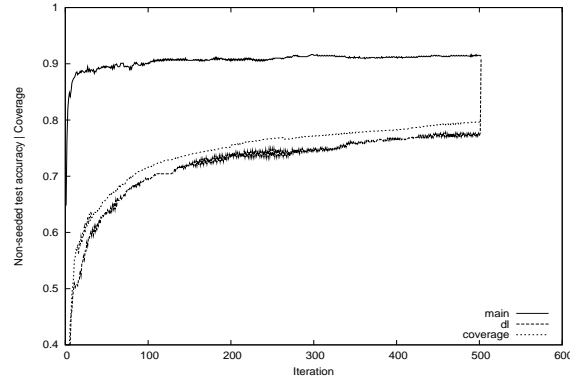
## 7 Conclusions

Our novel algorithm achieves accuracy comparable to Yarowsky-cautious, but is better theoretically motivated by combining ideas from Haffari and Sarkar (2007) and Subramanya et al. (2010). It also achieves accuracy comparable to DL-CoTrain, but does not require the features to be split into two independent views.

As future work, we would like to apply our al-



(a) Non-cautious



(b) Cautious

Figure 5: Internal train set coverage and non-seeded test accuracy (same scale) for Yarowsky-prop  $\theta$ -only on named entity.

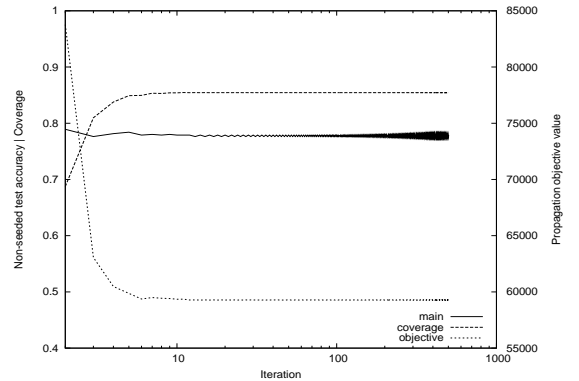


Figure 6: Non-seeded test accuracy (left axis), coverage (left axis, same scale), and objective value (right axis) for Yarowsky-prop  $\phi$ - $\theta$ . Iterations are shown on a log scale. We omit the first iteration (where the DL contains only the seed rules) and start the plot at iteration 2 where there is a complete DL.

gorithm to a structured task such as part of speech tagging. We also believe that our method for adapting Collins and Singer (1999)'s cautiousness to Yarowsky-prop can be applied to similar algorithms with other underlying classifiers, even to structured output models such as conditional random fields.

## References

- S. Abney. 2004. Understanding the Yarowsky algorithm. *Computational Linguistics*, 30(3).
- Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, DL '00.
- A. Blum and S. Chawla. 2001. Learning from labeled and unlabeled data using graph mincuts. In *Proc. 19th International Conference on Machine Learning (ICML-2001)*.
- A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of Computational Learning Theory*.
- Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *EMNLP 1999: Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–110.
- Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain, July. Association for Computational Linguistics.
- Hal Daume. 2011. Seeding, transduction, out-of-sample error and the Microsoft approach... Blog post at <http://nlpers.blogspot.com/2011/04/seeding-transduction-out-of-sample.html>, April 6.
- Jason Eisner and Damianos Karakos. 2005. Bootstrapping without the boot. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 395–402, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.
- Gholamreza Haffari and Anoop Sarkar. 2007. Analysis of semi-supervised learning with the Yarowsky algorithm. In *UAI 2007, Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence, Vancouver, BC, Canada*, pages 159–166.
- Aria Haghighi and Dan Klein. 2006a. Prototype-driven grammar induction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 881–888, Sydney, Australia, July. Association for Computational Linguistics.
- Aria Haghighi and Dan Klein. 2006b. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA, June. Association for Computational Linguistics.
- Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics - Volume 2, COLING '92*, pages 539–545, Stroudsburg, PA, USA. Association for Computational Linguistics.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning, ICML '01*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 30(3).
- Ellen Riloff and Jessica 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.
- H. J. Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11:363–371.
- Amarnag Subramanya, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 167–176, Cambridge, MA, October. Association for Computational Linguistics.
- Partha Pratim Talukdar. 2010. *Graph-based weakly-supervised methods for information extraction & integration*. Ph.D. thesis, University of Pennsylvania. Software: <https://github.com/parthatalukdar/junto>.
- X. Zhu and Z. Ghahramani and J. Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of International Conference on Machine Learning*.
- David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 189–196, Cambridge, Massachusetts, USA, June. Association for Computational Linguistics.



# Selective Sharing for Multilingual Dependency Parsing

**Tahira Naseem**  
CSAIL, MIT  
tahira@csail.mit.edu

**Regina Barzilay**  
CSAIL, MIT  
regina@csail.mit.edu

**Amir Globerson**  
Hebrew University  
gamir@cs.huji.ac.il

## Abstract

We present a novel algorithm for multilingual dependency parsing that uses annotations from a diverse set of source languages to parse a new unannotated language. Our motivation is to broaden the advantages of multilingual learning to languages that exhibit significant differences from existing resource-rich languages. The algorithm learns which aspects of the source languages are relevant for the target language and ties model parameters accordingly. The model factorizes the process of generating a dependency tree into two steps: *selection* of syntactic dependents and their *ordering*. Being largely language-universal, the selection component is learned in a supervised fashion from all the training languages. In contrast, the ordering decisions are only influenced by languages with similar properties. We systematically model this cross-lingual sharing using typological features. In our experiments, the model consistently outperforms a state-of-the-art multilingual parser. The largest improvement is achieved on the non Indo-European languages yielding a gain of 14.4%.<sup>1</sup>

## 1 Introduction

Current top performing parsing algorithms rely on the availability of annotated data for learning the syntactic structure of a language. Standard approaches for extending these techniques to resource-lean languages either use parallel corpora or rely on

annotated trees from other source languages. These techniques have been shown to work well for language families with many annotated resources (such as Indo-European languages). Unfortunately, for many languages there are no available parallel corpora or annotated resources in related languages. For such languages the only remaining option is to resort to unsupervised approaches, which are known to produce highly inaccurate results.

In this paper, we present a new multilingual algorithm for dependency parsing. In contrast to previous approaches, this algorithm can learn dependency structures using annotations from a diverse set of source languages, even if this set is not related to the target language. In our *selective sharing* approach, the algorithm learns which aspects of the source languages are relevant for the target language and ties model parameters accordingly. This approach is rooted in linguistic theory that characterizes the connection between languages at various levels of sharing. Some syntactic properties are universal across languages. For instance, nouns take adjectives and determiners as dependents, but not adverbs. However, the order of these dependents with respect to the parent is influenced by the typological features of each language.

To implement this intuition, we factorize generation of a dependency tree into two processes: selection of syntactic dependents and their ordering. The first component models the distribution of dependents for each part-of-speech tag, abstracting over their order. Being largely language-universal, this distribution can be learned in a supervised fashion from all the training languages. On the other hand,

<sup>1</sup>The source code for the work presented in this paper is available at <http://groups.csail.mit.edu/rbg/code/unidep/>

ordering of dependents varies greatly across languages and therefore should only be influenced by languages with similar properties. Furthermore, this similarity has to be expressed at the level of dependency types – i.e., two languages may share noun-adposition ordering, but differ in noun-determiner ordering. To systematically model this cross-lingual sharing, we rely on typological features that reflect ordering preferences of a given language. In addition to the known typological features, our parsing model embeds latent features that can capture cross-lingual structural similarities.

While the approach described so far supports a seamless transfer of shared information, it does not account for syntactic properties of the target language unseen in the training languages. For instance, in the CoNLL data, Arabic is the only language with the VSO ordering. To handle such cases, our approach augments cross-lingual sharing with unsupervised learning on the target languages.

We evaluated our selective sharing model on 17 languages from 10 language families. On this diverse set, our model consistently outperforms state-of-the-art multilingual dependency parsers. Performance gain, averaged over all the languages, is 5.9% when compared to the highest baseline. Our model achieves the most significant gains on non-Indo-European languages, where we see a 14.4% improvement. We also demonstrate that in the absence of observed typological information, a set of automatically induced latent features can effectively work as a proxy for typology.

## 2 Related Work

Traditionally, parallel corpora have been a mainstay of multilingual parsing (Wu, 1997; Kuhn, 2004; Smith and Smith, 2004; Hwa et al., 2005; Xi and Hwa, 2005; Burkett and Klein, 2008; Snyder et al., 2009). However, recent work in multilingual parsing has demonstrated the feasibility of transfer in the absence of parallel data. As a main source of guidance, these methods rely on the commonalities in dependency structure across languages. For instance, Naseem et al. (2010) explicitly encode these similarities in the form of universal rules which guide grammar induction in the target language. An alternative approach is to directly employ a non-lexicalized

parser trained on one language to process a target language (Zeman and Resnik, 2008; McDonald et al., 2011; Søgaard, 2011). Since many unlexicalized dependencies are preserved across languages, these approaches are shown to be effective for related languages. For instance, when applied to the language pairs within the Indo-European family, such parsers outperform unsupervised monolingual techniques by a significant margin.

The challenge, however, is to enable dependency transfer for target languages that exhibit structural differences from source languages. In such cases, the extent of multilingual transfer is determined by the relation between source and target languages. Berg-Kirkpatrick and Klein (2010) define such a relation in terms of phylogenetic trees, and use this distance to selectively tie the parameters of monolingual syntactic models. Cohen et al. (2011) do not use a predefined linguistic hierarchy of language relations, but instead learn the contribution of source languages to the training mixture based on the likelihood of the target language. Søgaard (2011) proposes a different measure of language relatedness based on perplexity between POS sequences of source and target languages. Using this measure, he selects a subset of training source sentences that are closer to the target language. While all of the above techniques demonstrate gains from modeling language relatedness, they still underperform when the source and target languages are unrelated.

Our model differs from the above approaches in its emphasis on the selective information sharing driven by language relatedness. This is further combined with monolingual unsupervised learning. As our evaluation demonstrates, this layered approach broadens the advantages of multilingual learning to languages that exhibit significant differences from the languages in the training mix.

## 3 Linguistic Motivation

### Language-Independent Dependency Properties

Despite significant syntactic differences, human languages exhibit striking similarity in dependency patterns. For a given part-of-speech tag, the set of tags that can occur as its dependents is largely consistent across languages. For instance, adverbs and nouns are likely to be dependents of verbs, while adjectives

are not. Thus, these patterns can be freely transferred across languages.

**Shared Dependency Properties** Unlike dependent selection, the ordering of dependents in a sentence differs greatly across languages. In fact, cross-lingual syntactic variations are primarily expressed in different ordering of dependents (Harris, 1968; Greenberg, 1963). Fortunately, the dimensions of these variations have been extensively studied in linguistics and are documented in the form of typological features (Comrie, 1989; Haspelmath et al., 2005). For instance, most languages are either dominantly prepositional like English or post-positional like Urdu. Moreover, a language may be close to different languages for different dependency types. For instance, Portuguese is a prepositional language like English, but the order of its noun-adjective dependency is different from English and matches that of Arabic. Therefore, we seek a model that can express parameter sharing at the level of dependency types and can benefit from known language relations.

**Language-specific Dependency Variations** Not every aspect of syntactic structure is shared across languages. This is particularly true given a limited number of supervised source languages; it is quite likely that a target language will have previously unseen syntactic phenomena. In such a scenario, the raw text in the target language might be the only source of information about its unique aspects.

## 4 Model

We propose a probabilistic model for generating dependency trees that facilitates parameter sharing across languages. We assume a setup where dependency tree annotations are available for a set of source languages and we want to use these annotations to infer a parser for a target language. Syntactic trees for the target language are not available during training. We also assume that both source and target languages are annotated with a coarse parts-of-speech tagset which is shared across languages. Such tagsets are commonly used in multilingual parsing (Zeman and Resnik, 2008; McDonald et al., 2011; Søgaard, 2011; Naseem et al., 2010).

The key feature of our model is a two-tier approach that separates the *selection* of dependents from their *ordering*:

1. *Selection Component*: Determines the dependent tags given the parent tag.
2. *Ordering Component*: Determines the position of each dependent tag with respect to its parent (right or left) and the order within the right and left dependents.

This factorization constitutes a departure from traditional parsing models where these decisions are tightly coupled. By separating the two, the model is able to support different degrees of cross-lingual sharing on each level.

For the selection component, a reasonable approximation is to assume that it is the same for all languages. This is the approach we take here.

As mentioned in Section 3, the ordering of dependents is largely determined by the typological features of the language. We assume that we have a set of such features for every language  $l$ , and denote this feature vector by  $v_l$ . We also experiment with a variant of our model where typological features are not observed. Instead, the model captures structural variations across languages by means of a small set of binary latent features. The values of these features are language dependent. We denote the set of latent features for language  $l$  by  $b_l$ .

Finally, based on the well known fact that long distance dependencies are less likely (Eisner and Smith, 2010), we bias our model towards short dependencies. This is done by imposing a corpus-level soft constraint on dependency lengths using the posterior regularization framework (Graça et al., 2007).

### 4.1 Generative Process

Our model generates dependency trees one fragment at a time. A *fragment* is defined as a subtree comprising the immediate dependents of any node in the tree. The process recursively generates fragments in a head outwards manner, where the distribution over fragments depends on the head tag. If the generated fragment is not empty then the process continues for each child tag in the fragment, drawing new fragments from the distribution associated with the tag. The process stops when there are no more non-empty fragments.

A fragment with head node  $h$  is generated in language  $l$  via the following stages:

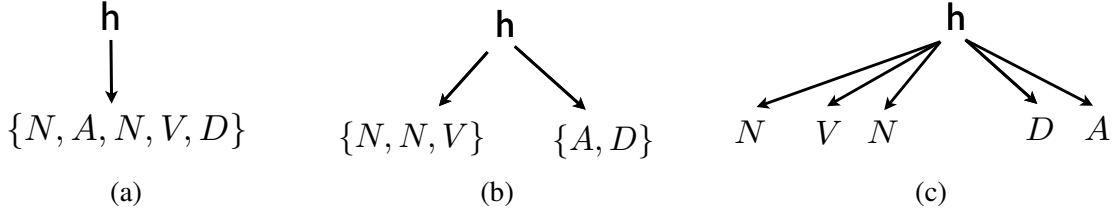


Figure 1: The steps of the generative process for a fragment with head  $h$ . In step (a), the unordered set of dependents is chosen. In step (b) they are partitioned into left and right unordered sets. Finally, each set is ordered in step (c).

- Generate the set of dependents of  $h$  via a distribution  $P_{sel}(S|h)$ . Here  $S$  is an unordered set of POS tags. Note that this part is universal (i.e., it does not depend on the language  $l$ ).
- For each element in  $S$  decide whether it should go to the right or left of  $h$  as follows: for every  $a \in S$ , draw its direction from the distribution  $P_{ord}(d|a, h, l)$ , where  $d \in \{R, L\}$ . This results in two unordered sets  $S_R, S_L$ , the right and left dependents of  $h$ . This part *does* depend on the language  $l$ , since the relative ordering of dependents is not likely to be universal.
- Order the sets  $S_R, S_L$ . For simplicity, we assume that the order is drawn uniformly from all the possible unique permutations over  $S_R$  and  $S_L$ . We denote the number of such unique permutations of  $S_R$  by  $n(S_R)$ .<sup>2</sup> Thus the probability of each permutation of  $S_R$  is  $\frac{1}{n(S_R)}$ .<sup>3</sup>

Figure 1 illustrates the generative process. The first step constitutes the *selection* component and the last two steps constitute the *ordering* component. Given this generation scheme, the probability  $P(D)$  of generating a given fragment  $D$  with head  $h$  will be:

$$P_{sel}(\{D\}|h) \prod_{a \in D} P_{ord}(d_D(a)|a, h, l) \frac{1}{n(D_R)n(D_L)} \quad (1)$$

Where we use the following notations:

- $D_R, D_L$  denote the parts of the fragment that are to the left and right of  $h$ .

<sup>2</sup>This number depends on the count of each distinct tag in  $S_R$ . For example if  $S_R = \{N, N, N\}$  then  $n(S_R) = 1$ . If  $S_R = \{N, D, V\}$  then  $n(S_R) = 3!$ .

<sup>3</sup>We acknowledge that assuming a uniform distribution over the permutations of the right and left dependents is linguistically counterintuitive. However, it simplifies the model by greatly reducing the number of parameters to learn.

- $\{D\}$  is the unordered set of tags in  $D$ .
- $d_D(a)$  is the position (either  $R$  or  $L$ ) of the dependent  $a$  w.r.t. the head of  $D$ .

In what follows we discuss the parameterizations of the different distributions.

#### 4.1.1 Selection Component

The selection component draws an unordered set of tags  $S$  given the head tag  $h$ . We assume that the process is carried out in two steps. First the number of dependents  $n$  is drawn from a distribution:

$$P_{size}(n|h) = \theta_{size}(n|h) \quad (2)$$

where  $\theta_{size}(n|h)$  is a parameter for each value of  $n$  and  $h$ . We restrict the maximum value of  $n$  to four, since this is a reasonable bound on the total number of dependents for a single parent node in a tree. These parameters are non-negative and satisfy  $\sum_n \theta_{size}(n|h) = 1$ . In other words, the size is drawn from a categorical distribution that is fully parameterized.

Next, given the size  $n$ , a set  $S$  with  $|S| = n$  is drawn according to the following log-linear model:

$$P_{set}(S|h, n) = \frac{1}{Z_{set}(h, n)} e^{\sum_{S_i \in S} \theta_{sel}(S_i|h)}$$

$$Z_{set}(h, n) = \sum_{S: |S|=n} e^{\sum_{S_i \in S} \theta_{sel}(S_i|h)}$$

In the above,  $S_i$  is the  $i^{th}$  POS tag in the unordered set  $S$ , and  $\theta_{sel}(S_i|h)$  are parameters. Thus, large values of  $\theta_{sel}(S_i|h)$  indicate that POS  $S_i$  is more likely to appear in the subset with parent POS  $h$ .

Combining the above two steps we have the following distribution for selecting a set  $S$  of size  $n$ :

$$P_{sel}(S|h) = P_{size}(n|h) P_{set}(S|h, n) . \quad (3)$$

ID	Feature Description	Values
81A	Order of Subject, Object and Verb	SVO, SOV, VSO, VOS, OVS, OSV
85A	Order of Adposition and Noun	Postpositions, Prepositions, Inpositions
86A	Order of Genitive and Noun	Genitive-Noun, Noun-Genitive
87A	Order of Adjective and Noun	Adjective-Noun, Noun-Adjective
88A	Order of Demonstrative and Noun	Demonstrative-Noun, Noun-Demonstrative
89A	Order of Numeral and Noun	Numeral-Noun, Noun-Numeral

Table 1: The set of typological features that we use in our model. For each feature, the first column gives the ID of the feature as used in WALS, the second column describes the feature and the last column enumerates the allowable values for the feature. Besides these values, each feature can also have a value of ‘No dominant order’.

#### 4.1.2 Ordering Component

The ordering component consists of distributions  $P_{ord}(d|a, h, l)$  that determine whether tag  $a$  will be mapped to the left or right of the head tag  $h$ . We model it using the following log-linear model:

$$P_{ord}(d|a, h, l) = \frac{1}{Z_{ord}(a, h, l)} e^{\mathbf{w}_{ord} \cdot \mathbf{g}(d, a, h, \mathbf{v}_l)}$$

$$Z_{ord}(a, h, l) = \sum_{d \in \{R, L\}} e^{\mathbf{w}_{ord} \cdot \mathbf{g}(d, a, h, \mathbf{v}_l)}$$

Note that in the above equations the ordering component depends on the known typological features  $\mathbf{v}_l$ . In the setup when typological features are not known,  $\mathbf{v}_l$  is replaced with the latent ordering feature set  $\mathbf{b}_l$ .

The feature vector  $\mathbf{g}$  contains indicator features for combinations of  $a, h, d$  and individual features  $v_{li}$  (i.e., the  $i^{th}$  typological features for language  $l$ ).

#### 4.2 Typological Features

The typological features we use are a subset of order-related typological features from ‘‘The World Atlas of Language Structure’’ (Haspelmath et al., 2005). We include only those features whose values are available for all the languages in our dataset. Table 1 summarizes the set of features that we use. Note that we do not explicitly specify the correspondence between these features and the model parameters. Instead, we leave it for the model to learn this correspondence automatically.

#### 4.3 Dependency Length Constraint

To incorporate the intuition that long distance dependencies are less likely, we impose a posterior constraint on dependency length. In particular, we use the Posterior Regularization (PR) framework of Graça et al. (2007). The PR framework incorporates

constraints by adding a penalty term to the standard likelihood objective. This term penalizes the distance of the model posterior from a set  $\mathcal{Q}$ , where  $\mathcal{Q}$  contains all the posterior distributions that satisfy the constraints. In our case the constraint is that the expected dependency length is less than or equal to a pre-specified threshold value  $b$ . If we denote the latent dependency trees by  $z$  and the observed sentences by  $x$  then

$$\mathcal{Q} = \{q(z|x) : E_q[f(x, z)] \leq b\} \quad (4)$$

where  $f(x, z)$  computes the sum of the lengths of all dependencies in  $z$  with respect to the linear order of  $x$ . We measure the length of a dependency relation by counting the number of tokens between the head and its modifier. The PR objective penalizes the KL-divergence of the model posterior from the set  $\mathcal{Q}$ :

$$\mathcal{L}_\theta(x) - \text{KL}(\mathcal{Q} \parallel p_\theta(z|x))$$

where  $\theta$  denotes the model parameters and the first term is the log-likelihood of the data. This objective can be optimized using a modified version of the EM algorithm (Graça et al., 2007).

### 5 Parameter Learning

Our model is parameterized by the parameters  $\theta_{sel}$ ,  $\theta_{size}$  and  $\mathbf{w}_{ord}$ . We learn these by maximizing the likelihood of the training data. As is standard, we add  $\ell_2$  regularization on the parameters and tune it on source languages. The likelihood is marginalized over all latent variables. These are:

- For sentences in the target language: all possible derivations that result in the observed POS tag sequences. The derivations include the choice of unordered sets size  $n$ , the unordered sets themselves  $S$ , their left/right al-

locations and the orderings within the left and right branches.

- For all languages: all possible values of the latent features  $\mathbf{b}_l$ .<sup>4</sup>

Since we are learning with latent variables, we use the EM algorithm to monotonically improve the likelihood. At each E step, the posterior over latent variables is calculated using the current model. At the M step this posterior is used to maximize the likelihood over the fully observed data. To compensate for the differences in the amount of training data, the counts from each language are normalized before computing the likelihood.

The M step involves finding maximum likelihood parameters for log-linear models in Equations 3 and 4. This is done via standard gradient based search; in particular, we use the method of BFGS.

We now briefly discuss how to calculate the posterior probabilities. For estimating the  $\mathbf{w}_{ord}$  parameters we require marginals of the type  $P(b_{li}|\mathcal{D}_l; \mathbf{w}^t)$  where  $\mathcal{D}_l$  are the sentences in language  $l$ ,  $b_{li}$  is the  $i_{th}$  latent feature for the language  $l$  and  $\mathbf{w}^t$  are the parameter values at iteration  $t$ . Consider doing this for a source language  $l$ . Since the parses are known, we only need to marginalize over the other latent features. This can be done in a straightforward manner by using our probabilistic model. The complexity is exponential in the number of latent features, since we need to marginalize over all features other than  $b_{li}$ . This is feasible in our case, since we use a relatively small number of such features.

When performing unsupervised learning for the target language, we need to marginalize over possible derivations. Specifically, for the M step, we need probabilities of the form  $P(a \text{ modifies } h|\mathcal{D}_l; \mathbf{w}^t)$ . These can be calculated using a variant of the inside outside algorithm. The exact version of this algorithm would be exponential in the number of dependents due to the  $\frac{1}{n(S_r)}$  term in the permutation factor. Although it is possible to run this exact algorithm in our case, where the number of dependents is limited to 4, we use an approximation that works well in practice: instead of  $\frac{1}{n(S_r)}$  we use  $\frac{1}{|S_r|!}$ . In this case the runtime is no longer exponential in the number of children, so inference is much faster.

<sup>4</sup>This corresponds to the case when typological features are not known.

Finally, given the trained parameters we generate parses in the target language by calculating the maximum a posteriori derivation. This is done using a variant of the CKY algorithm.

## 6 Experimental Setup

**Datasets and Evaluation** We test the effectiveness of our approach on 17 languages: Arabic, Basque, Bulgarian, Catalan, Chinese, Czech, Dutch, English, German, Greek, Hungarian, Italian, Japanese, Portuguese, Spanish, Swedish and Turkish. We used datasets distributed for the 2006 and 2007 CoNLL Shared Tasks (Buchholz and Marsi, 2006; Nivre et al., 2007). Each dataset provides manually annotated dependency trees and POS tags. To enable crosslingual sharing, we map the gold part-of-speech tags in each corpus to a common coarse tagset (Zeman and Resnik, 2008; Sjøgaard, 2011; McDonald et al., 2011; Naseem et al., 2010). The coarse tagset consists of 11 tags: noun, verb, adjective, adverb, pronoun, determiner, adposition, numeral, conjunction, particle, punctuation mark, and X (a catch-all tag). Among several available fine-to-coarse mapping schemes, we employ the one of Naseem et al. (2010) that yields consistently better performance for our method and the baselines than the mapping proposed by Petrov et al. (2011).

As the evaluation metric, we use directed dependency accuracy. Following standard evaluation practices, we do not evaluate on punctuation. For both the baselines and our model we evaluate on all sentences of length 50 or less ignoring punctuation.

**Training Regime** Our model typically converges quickly and does not require more than 50 iterations of EM. When the model involves latent typological variables, the initialization of these variables can impact the final performance. As a selection criterion for initialization, we consider the performance of the final model averaged over the supervised source languages. We perform ten random restarts and select the best according to this criterion. Likewise, the threshold value  $b$  for the PR constraint on the dependency length is tuned on the source languages, using average test set accuracy as the selection criterion.

**Baselines** We compare against the state-of-the-art multilingual dependency parsers that do not use parallel corpora for training. All the systems were eval-

uated using the same fine-to-coarse tagset mapping. The first baseline, *Transfer*, uses direct transfer of a discriminative parser trained on all the source languages (McDonald et al., 2011). This simple baseline achieves surprisingly good results, within less than 3% difference from a parser trained using parallel data. In the second baseline (*Mixture*), parameters of the target language are estimated as a weighted mixture of the parameters learned from annotated source languages (Cohen et al., 2011). The underlying parsing model is the dependency model with valance (DMV) (Klein and Manning, 2004). Originally, the baseline methods were evaluated on different sets of languages using a different tag mapping. Therefore, we obtained new results for these methods in our setup. For the *Transfer* baseline, for each target language we trained the model on all other languages in our dataset. For the *Mixture* baseline, we trained the model on the same four languages used in the original paper — English, German, Czech and Italian. When measuring the performance on these languages, we selected another set of four languages with a similar level of diversity.<sup>5</sup>

## 7 Results

Table 2 summarizes the performance for different configurations of our model and the baselines.

**Comparison against Baselines** On average, the selective sharing model outperforms both baselines, yielding 8.9% gain over the weighted mixture model (Cohen et al., 2011) and 5.9% gain over the direct transfer method (McDonald et al., 2011). Our model outperforms the weighted mixture model on 15 of the 17 languages and the transfer method on 12 of the 17 languages. Most of the gains are obtained on non-Indo-European languages, that have little similarity with the source languages. For this set, the average gain over the transfer baseline is 14.4%. With some languages, such as Japanese, achieving gains of as much as 30%.

On Indo-European languages, the model performance is almost equivalent to that of the best performing baseline. To explain this result we con-

<sup>5</sup>We also experimented with a version of the Cohen et al. (2011) model trained on all the source languages. This setup resulted in decreased performance. For this reason, we chose to train the model on the four languages.

sider the performance of the supervised version of our model which constitutes an upper bound on the performance. The average accuracy of our supervised model on these languages is 66.8%, compared to the 76.3% of the unlexicalized MST parser. Since Indo-European languages are overrepresented in our dataset, a target language from this family is likely to exhibit more similarity to the training data. When such similarity is substantial, the transfer baseline will benefit from the power of a context-rich discriminative parser.

A similar trait can be seen by comparing the performance of our model to an oracle version of our model which selects the optimal source language for a given target language (column 7). Overall, our method performs similarly to this oracle variant. However, the gain for non Indo-European languages is 1.9% vs -1.3% for Indo-European languages.

**Analysis of Model Properties** We first test our hypothesis about the universal nature of the dependent selection. We compare the performance of our model (column 6) against a variant (column 8) where this component is trained from annotations on the target language. The performance of the two is very close – 1.8%, supporting the above hypothesis.

To assess the contribution of other layers of selective sharing, we first explore the role of typological features in learning the ordering component. When the model does not have access to observed typological features, and does not use latent ones (column 4), the accuracy drops by 2.6%<sup>6</sup>. For some languages (e.g., Turkish) the decrease is very pronounced. Latent typological features (column 5) do not yield the same gain as observed ones, but they do improve the performance of the typology-free model by 1.4%.

Next, we show the importance of using raw target language data in training the model. When the model has to make all the ordering decisions based on meta-linguistic features without account for unique properties of the target languages, the performance decreases by 0.9% (see column 3).

To assess the relative difficulty of learning the ordering and selection components, we consider model variants where each of these components is

<sup>6</sup>In this setup, the ordering component is trained in an unsupervised fashion on the target language.

	Baselines		Selective Sharing Model							
	Mixture	Transfer	(D-,T <sub>o</sub> )	(D+)	(D+,T <sub>l</sub> )	(D+,T <sub>o</sub> )	Best Pair	Sup. Sel.	Sup. Ord.	MLE
Catalan	64.9	69.5	71.9	66.1	66.7	71.8	74.8	70.2	73.2	72.1
Italian	61.9	68.3	68.0	65.5	64.2	65.6	68.3	65.1	70.7	72.3
Portuguese	72.9	75.8	76.2	72.3	76.0	73.5	76.4	77.4	77.6	79.6
Spanish	57.2	65.9	62.3	58.5	59.4	62.1	63.4	61.5	62.6	65.3
Dutch	50.1	53.9	56.2	56.1	55.8	55.9	57.8	56.3	58.6	58.0
English	45.9	47.0	47.6	48.5	48.1	48.6	44.4	46.3	60.0	62.7
German	54.5	56.4	54.0	53.5	54.3	53.7	54.8	52.4	56.2	58.0
Swedish	56.4	63.6	52.0	61.4	60.6	61.5	63.5	67.9	67.1	73.0
Bulgarian	67.7	64.0	67.6	63.5	63.9	66.8	66.1	66.2	69.5	71.0
Czech	39.6	40.3	43.9	44.7	45.4	44.6	47.5	53.2	51.2	58.9
Arabic	44.8	40.7	57.2	58.8	60.3	58.9	57.6	62.9	61.9	64.2
Basque	32.8	32.4	39.7	40.1	39.8	47.6	42.0	46.2	47.9	51.6
Chinese	46.7	49.3	59.9	52.2	52.0	51.2	65.4	62.3	65.5	73.5
Greek	56.8	60.4	61.9	67.5	67.3	67.4	60.6	67.2	69.0	70.5
Hungarian	46.8	54.3	56.9	58.4	58.8	58.5	57.0	57.4	62.0	61.6
Japanese	33.5	34.7	62.3	56.8	61.4	64.0	54.8	63.4	69.7	75.6
Turkish	28.3	34.3	59.1	43.6	57.8	59.2	56.9	66.6	59.5	67.6
Average	50.6	53.6	58.6	56.9	58.3	<b>59.5</b>	59.5	61.3	63.7	66.8

Table 2: Directed dependency accuracy of different variants of our selective sharing model and the baselines. The first section of the table (column 1 and 2) shows the accuracy of the weighted mixture baseline (Cohen et al., 2011) (Mixture) and the multi-source transfer baseline (McDonald et al., 2011) (Transfer). The middle section shows the performance of our model in different settings. D± indicates the presence/absence of raw target language data during training. T<sub>o</sub> indicates the use of observed typological features for all languages and T<sub>l</sub> indicates the use of latent typological features for all languages. The last section shows results of our model with different levels of oracle supervision: a. (Best Pair) Model parameters are borrowed from the best source language based on the accuracy on the target language b. (Sup. Sel.) Selection component is trained using MLE estimates from target language c. (Sup. Ord.) Ordering component is trained using MLE estimates from the target language d. (MLE) All model parameters are trained on the target language in a supervised fashion. The horizontal partitions separate language families. The first three families are sub-divisions of the Indo-European language family.

trained using annotations in the target language. As shown in columns 8 and 9, these two variants outperform the original model, achieving 61.3% for supervised selection and 63.7% for supervised ordering. Comparing these numbers to the accuracy of the original model (column 6) demonstrates the difficulty inherent in learning the ordering information. This finding is expected given that ordering involves selective sharing from multiple languages.

Overall, the performance gap between the selective sharing model and its monolingual supervised counterpart is 7.3%. In contrast, the unsupervised monolingual variant of our model achieves a meager 26%.<sup>7</sup> This demonstrates that our model can effectively learn relevant aspects of syntactic structure from a diverse set of languages.

<sup>7</sup>This performance is comparable to other generative models such as DMV (Klein and Manning, 2004).

## 8 Conclusions

We present a novel algorithm for multilingual dependency parsing that uses annotations from a diverse set of source languages to parse a new unannotated language. Overall, our model consistently outperforms the multi-source transfer based dependency parser of McDonald et al. (2011). Our experiments demonstrate that the model is particularly effective in processing languages that exhibit significant differences from the training languages.

## Acknowledgments

The authors acknowledge the support of the NSF (IIS-0835445), the MURI program (W911NF-10-1-0533), the DARPA BOLT program, and the ISF (1789/11). We thank Tommi Jaakkola, Ryan McDonald and the members of the MIT NLP group for their comments.



## References

- Taylor Berg-Kirkpatrick and Dan Klein. 2010. Phylogenetic grammar induction. In *ACL*, pages 1288–1297.
- Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL*, pages 149–164.
- David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *Proceedings of EMNLP*, pages 877–886.
- Shay B. Cohen, Dipanjan Das, and Noah A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *EMNLP*, pages 50–61.
- Bernard Comrie. 1989. *Language Universals and Linguistic Typology: Syntax and Morphology*. Oxford: Blackwell.
- Jason Eisner and Noah A. Smith. 2010. Favor short dependencies: Parsing with soft and hard constraints on dependency length. In *Trends in Parsing Technology: Dependency Parsing, Domain Adaptation, and Deep Parsing*, pages 121–150.
- João Graça, Kuzman Ganchev, and Ben Taskar. 2007. Expectation maximization and posterior constraints. In *Advances in NIPS*, pages 569–576.
- Joseph H Greenberg. 1963. Some universals of language with special reference to the order of meaningful elements. In Joseph H Greenberg, editor, *Universals of Language*, pages 73–113. MIT Press.
- Z.S. Harris. 1968. *Mathematical structures of language*. Wiley.
- Martin Haspelmath, Matthew S. Dryer, David Gil, and Bernard Comrie, editors. 2005. *The World Atlas of Language Structures*. Oxford University Press.
- R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Journal of Natural Language Engineering*, 11(3):311–325.
- Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of ACL*, pages 478–485.
- Jonas Kuhn. 2004. Experiments in parallel-text based grammar induction. In *Proceedings of the ACL*, pages 470–477.
- Ryan T. McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *EMNLP*, pages 62–72.
- Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *EMNLP*, pages 1234–1244.
- Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.
- Slav Petrov, Dipanjan Das, and Ryan McDonald. 2011. A universal part-of-speech tagset. In *ArXiv*, April.
- David A. Smith and Noah A. Smith. 2004. Bilingual parsing with factored estimation: Using English to parse Korean. In *Proceeding of EMNLP*, pages 49–56.
- Benjamin Snyder, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *Proceedings of ACL/AFNLP*, pages 73–81.
- Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *ACL (Short Papers)*, pages 682–686.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.
- Chenhai Xi and Rebecca Hwa. 2005. A backoff model for bootstrapping resources for non-english languages. In *Proceedings of EMNLP*, pages 851 – 858.
- Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages*, pages 35–42, January.

# The Creation of a Corpus of English Metalanguage

Shomir Wilson\*

Carnegie Mellon University  
Pittsburgh, PA 15213, USA  
shomir@cs.cmu.edu

## Abstract

Metalanguage is an essential linguistic mechanism which allows us to communicate explicit information about language itself. However, it has been underexamined in research in language technologies, to the detriment of the performance of systems that could exploit it. This paper describes the creation of the first tagged and delineated corpus of English metalanguage, accompanied by an explicit definition and a rubric for identifying the phenomenon in text. This resource will provide a basis for further studies of metalanguage and enable its utilization in language technologies.

## 1 Introduction

In order to understand the language that we speak, we sometimes must refer to the language itself. Language users do this through an understanding of the *use-mention distinction*, as exhibited by the mechanism of *metalanguage*: that is, language that describes language. The use-mention distinction is illustrated simply in Sentences (1) and (2) below:

(1) I watch football on weekends.

(2) Football may refer to one of several sports.

A reader understands that *football* in Sentence (1) refers to a sporting activity, while the same word in Sentence (2) refers to the term *football* itself. Evidence suggests that human communication frequently employs metalanguage (Anderson et al. 2002), and the phenomenon is essential for many activities, including the introduction of new

vocabulary, attribution of statements, explanation of meaning, and assignment of names (Saka 2003). Sentences (3) through (8) below further illustrate the phenomenon, highlighted in bold.

(3) This is sometimes called **tough love**.

(4) I wrote “**meet outside**” on the chalkboard.

(5) **Has** is a conjugation of the verb **have**.

(6) The button labeled **go** was illuminated.

(7) That bus, was its name **61C**?

(8) **Mississippi** is fun to spell.

Recognizing a wide variety of metalinguistic constructions is a skill that humans take for granted in fellow interlocutors (Perlis, Purang & Andersen 1998), and it is a core language skill that children demonstrate at an early age (Clark & Schaefer 1989). Regardless of context, topic, or mode of communication (spoken or written), we are able to refer directly to language, and we expect others to recognize and understand when we do so.

The study of the syntax and semantics of metalanguage is well developed for formal languages. However, the study of the phenomenon in natural language is relatively nascent, and its incorporation into language technologies is almost non-existent. Parsing the distinction is difficult, as shown in Figure 1 below: *go* does not function as a verb in Sentence (6), but it is tagged as such. *Delineating* an instance of metalanguage with quotation marks is a common convention, but this often fails to ameliorate the parsing problem. Quotation marks, italic text, and bold text—three common conventions used to highlight metalanguage—are inconsistently applied and are already “overloaded” with several distinct uses.

Moreover, applications of natural language processing generally lack the ability to recognize and interpret metalanguage (Anderson et al. 2002).

---

\* This research was performed during a prior affiliation with the University of Maryland at College Park.

Systems using sentiment analysis are affected, as sentiment-suggestive terms appearing in metalanguage (especially in quotation, a form of the phenomenon (Maier 2007)) are not necessarily reflective of the writer or speaker. Applications of natural language understanding cannot process metalanguage without detecting it, especially when upstream components (such as parsers) mangle its structure. Interactive systems that could leverage users' expectations of metalanguage competency currently fail to do so. Figure 2 below shows a fragment of conversation with the *Let's Go!* (Raux et al. 2005) spoken dialog system, designed to help users plan trips on Pittsburgh's bus system.

```
(ROOT
  (S
    (NP
      (NP (DT The) (NN button))
      (VP (VBN labeled)
        (S
          (VP (VB go))))))
    (VP (VBD was)
      (VP (VBN illuminated)))
    (. .)))
```

Figure 1. Output of the Stanford Parser (Klein & Manning 2003) for Sentence (6). Adding quotation marks around *go* alters the parser output slightly (not shown), but *go* remains labeled VB.

```
Let's Go!: Where do you wish to depart
from?
User: Arlington.
Let's Go!: Departing from Allegheny
West. Is this right?
User: No, I said "Arlington".
Let's Go!: Please say where you are
leaving from.
```

Figure 2: A conversation with *Let's Go!* in which the user responds to a speech recognition error.

The exchange shown in Figure 2 is representative of the reactions of nearly all dialog systems: in spite of the domain generality of metalanguage and the user's expectation of its availability, the system does not recognize it and instead "talks past" the user. In effect, language technologies that ignore metalanguage are discarding the most direct source of linguistic information that text or utterances can provide.

This paper describes the first substantial study to characterize and gather instances of English metalanguage. Section 2 presents a definition and a rubric for metalanguage in the form of *mentioned language*. Section 3 describes the procedure used to create the corpus and some notable properties of its contents, and Section 4 discusses insights gained into the phenomenon. The remaining sections discuss the context of these results and future directions for this research.

## 2 Metalanguage and the Use-Mention Distinction<sup>1</sup>

Although the reader is likely to be familiar with the terms *use-mention distinction* and *metalanguage*, the topic merits further explanation to precisely establish the phenomenon being studied. Intuitively, the vast majority of utterances are produced for use rather than mention, as the roles of language-mention are auxiliary (albeit indispensable) to language use. This paper will adopt the term *mentioned language* to describe the literal, delineable phenomenon illustrated in examples thus far. Other forms of metalanguage occur through deictic references to linguistic entities that do not appear in the relevant statement. (For example, consider "That word was misspelled" where the referred-to word resides outside of the sentence.) For technical tractability, this study focuses on mentioned language.

### 2.1 Definition

Although the use-mention distinction has enjoyed a long history of theoretical discussion, attempts to explicitly define one or both of the distinction's disjuncts are difficult (or impossible) to find. Below is the definition of mentioned language adopted by this study, followed by clarifications.

*Definition: For T a token or a set of tokens in a sentence, if T is produced to draw attention to a property of the token T or the type of T, then T is an instance of mentioned language.*

Here, a *token* is the specific, situated (i.e., as appearing in the sentence) instantiation of a linguistic entity: a letter, symbol, sound, word, phrase, or another related entity. A *property* might

<sup>1</sup> The definition and rubric in this section were originally introduced by Wilson (2011a). For brevity, their full justifications and the argument for equivalence between the two are not reproduced here.

be a token's spelling, pronunciation, meaning (for a variety of interpretations of *meaning*), structure, connotation, original source (in cases of quotation), or another aspect for which language is shown or demonstrated. The *type* of T is relevant in most instances of mentioned language, but the *token* itself is relevant in others, as in the sentence below:

(9) "The" appears between quote marks here.

Constructions like (9) are unusual and are of limited practical value, but the definition accommodates them for completeness.

The adoption of this definition was motivated by a desire to study mentioned language with precise, repeatable results. However, it was too abstract to consistently apply to large quantities of candidate phrases in sentences, a necessity for corpus creation. A brief attempt to train annotators using the definition was unsuccessful, and instead a rubric was created for this purpose.

## 2.2 Annotation Rubric

A human reader with some knowledge of the use-mention distinction can often intuit the presence of mentioned language in a sentence. However, to operationalize the concept and move toward corpus construction, it was necessary to create a rubric for labeling it. The rubric is based on substitution, and it may be applied, with caveats described below, to determine whether a linguistic entity is mentioned by the sentence in which it occurs.

*Rubric: Suppose X is a linguistic entity in a sentence S. Construct sentence S' as follows: replace X in S with a phrase X' of the form "that [item]", where [item] is the appropriate term for X in the context of S (e.g., "letter", "symbol", "word", "name", "phrase", "sentence", etc.). X is an instance of mentioned language if, when assuming that X' refers to X, the meaning of S' is equivalent to the meaning of S.*

To further operationalize the rubric, Figure 3 shows it rewritten in pseudocode form. To verify the rubric, the reader can follow a positive example and a negative example in Figure 4.

To maintain coherency, minor adjustments in sentence wording will be necessary for some candidate phrases. For instance, Sentence (10) below must be rewritten as (11):

(10) The word *cat* is spelled with three letters.

(11) *Cat* is spelled with three letters.

This is because S' for (10) and (11) are respectively (12) and (13):

(12) The word that word is spelled with three letters.

(13) That word is spelled with three letters.

```

Given S a sentence and X a copy of a
linguistic entity in S:
(1) Create X': the phrase "that [item]",
    where [item] is the appropriate term
    for linguistic entity X in the
    context of S.
(2) Create S': copy S and replace the
    occurrence of X with X'.
(3) Create W: the set of truth
    conditions of S.
(4) Create W': the set of truth
    conditions of S', assuming that X'
    in S' is understood to refer
    deictically to X.
(5) Compare W and W'. If they are equal,
    X is mentioned language in S. Else,
    X is not mentioned language in S.

```

Figure 3: Pseudocode equivalent of the rubric.

```

Positive Example
S: Spain is the name of a European
country.
X: Spain.
(1) X': that name
(2) S': That name is the name of a
European country.
(3) W: Stated briefly, Spain is the name
of a European country.
(4) W': Stated briefly, Spain is the
name of a European country.
(5) W and W' are equal. Spain is
mentioned language in S.

Negative Example
S: Spain is a European country.
X: Spain.
(1) X': that name
(2) S': That name is a European country.
(3) W: Stated briefly, Spain is a
European country.
(4) W': Stated briefly, the name Spain
is a European country.
(5) W and W' are not equal. Spain is not
mentioned language in S.

```

Figure 4: Examples of rubric application using the pseudocode in Figure 3.

Also, quotation marks around or inside of a candidate phrase require special attention, since their inclusion or exclusion in X can alter the meaning of S'. For this discussion, quotation marks

and other stylistic cues are considered informal cues which aid a reader in detecting mentioned language. Style conventions may call for them, and in some cases they might be strictly necessary, but a competent language user possesses sufficient skill to properly discard or retain them as each instance requires (Saka 1998).

### 3 The Mentioned Language Corpus

“Laboratory examples” of mentioned language (such as the examples thus far in this paper) only begin to illustrate the variation in the phenomenon. To conduct an empirical examination of mentioned language and to study the feasibility of automatic identification, it was necessary to gather a large, diverse set of samples. This section describes the process of building a series of three progressively more sophisticated corpora of mentioned language. The first two were previously constructed by Wilson (2010; 2011b) and will be described only briefly. The third was built with insights from the first two, and it will be described in greater detail. This third corpus is the first to *delineate* mentioned language: that is, it identifies precise subsequences of words in a sentence that are subject to the phenomenon. Doing so will enable analysis of the syntax and semantics of English metalanguage.

#### 3.1 Approach

The article set of *English Wikipedia*<sup>2</sup> was chosen as a source for text, from which instances were mined using a combination of automated and manual efforts. Four factors led to its selection:

- 1) *Wikipedia is collaboratively written*. Since any registered user can contribute to articles, Wikipedia reflects the language habits of a large sample of English writers (Adler et al. 2008).
- 2) *Stylistic cues that sometimes delimit mentioned language are present in article text*. Contributors tend to use quote marks, italic text, or bold text to delimit mentioned language<sup>3</sup>, thus following conventions respected across many domains of writing (Strunk & White 1979; Chicago Editorial Staff 2010; American Psychological Association. 2001). Discussion

<sup>2</sup> Described in detail at

[http://en.wikipedia.org/wiki/English\\_Wikipedia](http://en.wikipedia.org/wiki/English_Wikipedia).

<sup>3</sup> These conventions are stated in Wikipedia’s style manual, though it is unclear whether most contributors read the manual or follow the conventions out of habit.

boards and other sources of informal language were considered, but the lack of consistent (or any) stylistic cues would have made candidate phrase collection untenably time-consuming.

- 3) *Articles are written to introduce a wide variety of concepts to the reader*. Articles are written informatively and they generally assume the reader is unfamiliar with their topics, leading to frequent instances of mentioned language.
- 4) *Wikipedia is freely available*. Various language learning materials were also considered, but legal and technical obstacles made them unsuitable for creating a freely available corpus.

To construct each of the three corpora, a general procedure was followed. First, a set of current article revisions was downloaded from Wikipedia. Then, the main bodies of article text (excluding discussion pages, image captions, and other peripheral text) were scanned for sentences that contained instances of highlighted text (i.e., text inside of the previously mentioned stylistic cues). Since stylistic cues are also used for other language tasks, candidate instances were heuristically filtered and then annotated by human readers.

#### 3.2 Previous Efforts

In previous work, a pilot corpus was constructed to verify the fertility of Wikipedia as a source for mentioned language. From 1,000 articles, 1,339 sentences that contained stylistic cues were examined by a human reader, and 171 were found to contain at least one instance of mentioned language. Although this effort verified Wikipedia’s viability for the project, it also revealed that the hand-labeling procedure was time-consuming, and prior heuristic filtering would be necessary.

Next, the “Combined Cues” corpus was constructed to test the combination of stylistic filtering and a new lexical filter for selecting candidate instances. A set of 23 “mention-significant” words was gathered informally from the pilot corpus, consisting of nouns and verbs:

*Nouns: letter, meaning, name, phrase, pronunciation, sentence, sound, symbol, term, title, word*

*Verbs: ask, call, hear, mean, name, pronounce, refer, say, tell, title, translate, write*

Instances of highlighted text were only promoted to the hand annotation stage if they contained at least one of these words within the three-word phrase directly preceding the

highlighted text. From 3,831 articles, a set of 898 sentences were found to contain 1,164 candidate instances that passed the combination of stylistic and lexical filters. Hand annotation of those candidates yielded 1,082 instances of mentioned language. Although the goal of the filters was only to ease hand annotation, it could be stated that the filters had almost 93% precision in detecting the phenomenon. It did not seem plausible that the set of mention-significant words was complete enough to justify that high percentage, and concerns were raised that the lexical filter was rejecting many instances of mentioned language.

### 3.3 The “Enhanced Cues” Corpus

The construction of the present corpus (referred to as the “Enhanced Cues” Corpus) was similar to previous efforts but used a much-enlarged set of mention-significant nouns and verbs gathered from the *WordNet* (Fellbaum 1998) lexical ontology. For each of the 23 original mention-significant words, a human reader started with its containing synset and followed hypernym links until a synset was reached that did not refer to a linguistic entity. Then, backtracking one synset, all lemmas of all descendants of the most general linguistically-relevant synset were gathered. Figure 5 illustrates this procedure with an example.

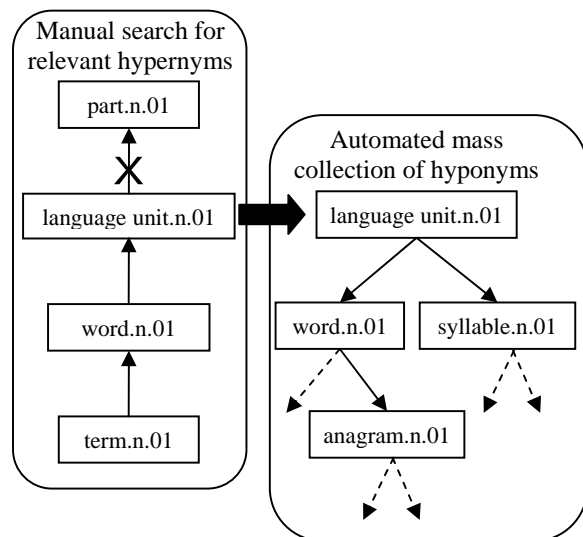


Figure 5: Gathering mention-significant words from WordNet using the seed noun “term”. Here, “Language unit”, “word”, “syllable”, “anagram”, and all their descendants are gathered.

Using the combination of stylistic and lexical cues, 2,393 candidate instances were collected, and the researcher used the rubric and definition from Section 2 to identify 629 instances of mentioned language<sup>4</sup>. The researcher also identified four categories of mentioned language based on the nature of the substitution phrase X’ specified by the rubric. These categories will be discussed in the following subsection. Figure 6 summarizes this procedure and the numeric outcomes.

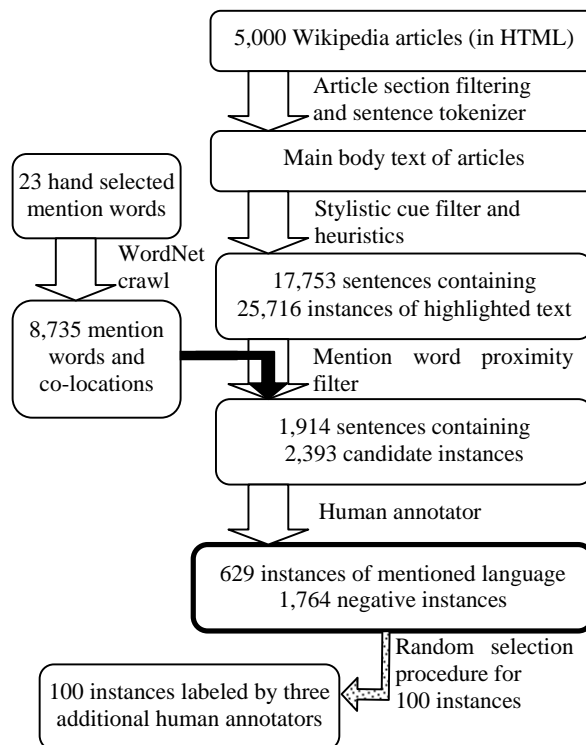


Figure 6: The procedure used to create the Enhanced Cues Corpus.

### 3.4 Corpus Composition

As stated previously, categories for mentioned language were identified based on intuitive relationships among the substitution phrases created for the rubric (e.g., “that word”, “that title”, “that symbol”). The categories are:

- 1) *Words as Words (WW)*: Within the context of the sentence, the candidate phrase is used to refer to the word or phrase itself and not what it usually refers to.

<sup>4</sup> This corpus is available at [http://www.cs.cmu.edu/~shomir/um\\_corpus.html](http://www.cs.cmu.edu/~shomir/um_corpus.html).

- 2) *Names as Names (NN)*: The sentence directly refers to the candidate phrase as a proper name, nickname, or title.
- 3) *Spelling or Pronunciation (SP)*: The candidate text appears only to illustrate spelling, pronunciation, or a character symbol.
- 4) *Other Mention/Interesting (OM)*: The candidate phrase is an instance of mentioned language that does not fit the above three categories.
- 5) *Not Mention (XX)*: The candidate phrase is not mentioned language.

Table 1 presents the frequencies of each category in the Enhanced Cues corpus, and Table 2 provides examples for each from the corpus. WW was by far the most common label to appear, which is perhaps an artifact of the use of Wikipedia as the text source. Although Wikipedia articles contain many names, NN was not as common, and informal observations suggested that names and titles are not as frequently introduced via metalanguage. Instead, their referents are introduced directly by the first appearance of the referring text. Spelling and pronunciation were particularly sparse; again, a different source might have yielded more examples for this category. The OM category was occupied mostly by instances of speech or language production by an agent, as illustrated by the two OM examples in Table 2.

Category	Code	Frequency
Words as Words	WW	438
Names as Names	NN	117
Spelling or Pronunciation	SP	48
Other Mention/Interesting	OM	26
Not Mention	XX	1,764

Table 1: The by-category composition of candidate instances in the Enhanced Cues corpus.

In the interest of revealing both lexical and syntactic cues for mentioned language, part-of-speech tags were computed (using NLTK (Loper & Bird 2002)) for words in all of the sentences containing candidate instances. Tables 3 and 4 list the ten most common words (as POS-tagged) in the three-word phrases before and after (respectively) candidate instances. Although the heuristics for collecting candidate instances were not intended to function as a classifier, figures for precision are shown for each word: these represent

the percentage of occurrences of the word which were associated with candidates identified as mentioned language. For example, 80% of appearances of the verb *call* preceded a candidate instance that was labeled as mentioned language.

Code	Example
WW	The IP Multimedia Subsystem architecture uses the term <u>transport plane</u> to describe a function roughly equivalent to the routing control plane.
NN	<u>Digeri</u> is the name of a Thracian tribe mentioned by Pliny the Elder, in The Natural History. Hazrat Syed Jalaluddin Bukhari's descendants are also called <u>Naqvi al-Bukhari</u> .
SP	The French changed the spelling to <u>bataillon</u> , whereupon it directly entered into German. Welles insisted on pronouncing the word apostles with a hard <u>t</u> .
OM	He kneels over Fil, and seeing that his <u>eyes are open whispers: brother</u> . During Christmas 1941, she typed <u>The end on the last page of Laura</u> .
XX	<u>NCR</u> was the first U.S. publication to write about the clergy sex abuse scandal. Many Croats reacted by <u>expelling</u> all words in the Croatian language that had, in their minds, even distant Serbian origin.

Table 2: Two examples from the corpus for each category. Candidate phrases appear underlined, with the original stylistic cues removed.

Many of these words appeared as mention words for the Combined Cues corpus, indicating that prior intuitions about framing metalanguage were correct. In particular, *call* (*v*), *word*(*n*), and *term* (*n*) were exceptionally frequent and effective at associating with mentioned language. In contrast, the distribution of frequencies for the words *following* candidate instances exhibited a “long tail”, indicating greater variation in vocabulary.

Rank	Word	Freq.	Precision (%)
1	call (v)	92	80
2	word (n)	68	95.8
3	term (n)	60	95.2
4	name (n)	31	67.4
5	use (v)	17	70.8
6	know (v)	15	88.2
7	also (rb)	13	59.1
8	name (v)	11	100
9	sometimes (rb)	9	81.9
10	Latin (n)	9	69.2

Table 3: The top ten words appearing in the three-word sequences *before* candidate instances, with precisions of association with mentioned language.

Rank	Word	Freq.	Precision (%)
1	mean (v)	31	83.4
2	name (n)	24	63.2
3	use (v)	11	55
4	meaning (n)	8	57.1
5	derive (v)	8	80
6	refers (n)	7	87.5
7	describe (v)	6	60
8	refer (v)	6	54.5
9	word (n)	6	50
10	may (md)	5	62.5

Table 4: The top ten words appearing in the three-word sequences *after* candidate instances, with precisions of association with mentioned language.

### 3.5 Reliability and Consistency of Annotation

To provide some indication of the reliability and consistency of the Enhanced Cues Corpus, three additional expert annotators were recruited to label a subset of the candidate instances. These additional annotators received guidelines for annotation that included the five categories, and they worked separately (from each other and from the primary annotator) to label 100 instances selected randomly with quotas for each category.

Calculations first were performed to determine the level of agreement on the mere presence of mentioned language, by mapping labels WW, NN, SP, and OM to *true* and XX to *false*. All four annotators agreed upon a *true* label for 46 instances and a *false* label for 30 instances, with an average pairwise Kappa (computed via NTLK) of 0.74. Kappa between the primary annotator and a

hypothetical “majority voter” of the three additional annotators was 0.90. These results were taken as moderate indication of the reliability of “simple” use-mention labeling.

However, the per-category results showed reduced levels of agreement. Kappa was calculated to be 0.61 for the original coding. Table 5 shows the Kappa statistic for binary re-mapping for each of the categories. This was done similarly to the “XX versus all others” procedure described above.

Code	Frequency	K
WW	17	0.38
NN	17	0.72
SP	16	0.66
OM	4	0.09
XX	46	0.74

Table 5: Frequencies of each category in the subset labeled by additional annotators and the values of the Kappa statistic for binary relabelings.

The low value for remapped OM was expected, since the category was small and intentionally not well-defined. The relatively low value for WW was not expected, though it seems possible that the redaction of specific stylistic cues made annotators less certain when to apply this category. Overall, these numbers suggest that, although annotators tend to agree whether a candidate instance is mentioned language or not, there is less of a consensus on how to qualify positive instances.

## 4 Discussion

The Enhanced Cues corpus confirms some of the hypothesized properties of metalanguage and yields some unexpected insights. Stylistic cues appear to be strongly associated with mentioned language; although the examination of candidate phrases was limited to “highlighted” text, informal perusal of the remainder of article text confirmed this association. Further evidence can be seen in examples from other texts, shown below with their original stylistic cues intact:

- Like so many words, the meaning of “addiction” has varied wildly over time, but the trajectory might surprise you.<sup>5</sup>

<sup>5</sup> News article from CNN.com:  
<http://www.cnn.com/2011/LIVING/03/23/addicted.to.addiction/index.html>



- Sending a signal in this way is called a **speech act**.<sup>6</sup>
- M1 and M2 are Slashdot shorthand for “moderation” and “metamoderation,” respectively.<sup>7</sup>
- He could explain foreordination thoroughly, and he used the terms “baptize” and “Athanasian.”<sup>8</sup>
- They use *Kabuki* precisely because they and everyone else have only a hazy idea of the word’s true meaning, and they can use it purely on the level of insinuation.<sup>9</sup>

However, the connection between mentioned language and stylistic cues is only valuable when stylistic cues are available. Still, even in their absence there appears to be an association between mentioned language and a core set of nouns and verbs. Recurring patterns were observed in how mention-significant words related to mentioned language. Two were particularly common:

- *Noun apposition between a mention-significant noun and mentioned language.* An example of this appears in Sentence (5), consisting of the noun *verb* and the mentioned word *have*.
- *Mentioned language appearing in appropriate semantic roles for mention-significant verbs.* Sentence (3) illustrates this, with the verb *call* assigning the label *tough love* as an attribute of the sentence subject.

With further study, it should be possible to exploit these relationships to automatically detect mentioned language in text.

## 5 Related Work

The use-mention distinction has enjoyed a long history of chiefly theoretical discussion. Beyond those authors already cited, many others have addressed it as the formal topic of *quotation* (Davidson 1979; Cappelen & Lepore 1997; García-Carpintero 2004; Partee 1973; Quine 1940; Tarski 1933). Nearly all of these studies have eschewed empirical treatments, instead hand-picking illustrations of the phenomenon.

<sup>6</sup> Page 684 of Russell and Norvig’s 1995 edition of *Artificial Intelligence*, a textbook.

<sup>7</sup> Frequently Asked Questions (FAQ) list on Slashdot.org: <http://slashdot.org/faq/metamod.shtml>

<sup>8</sup> Novel *Elmer Gantry* by Sinclair Lewis.

<sup>9</sup> Opinion column on Slate.com: <http://www.slate.com/id/2250081/>

One notable exception was a study by Anderson et al. (2004), who created a corpus of metalanguage from a subset of the British National Corpus, finding that approximately 11% of spoken utterances contained some form (whether explicit or implicit) of metalanguage. However, limitations in the Anderson corpus’ structure (particularly lack of word- or phrase-level annotations) and content (the authors admit it is noisy) served as compelling reasons to start afresh and create a richer resource.

## 6 Future Work

As explained in the introduction, the long-term goal of this research program is to apply an understanding of metalanguage to enhance language technologies. However, the more immediate goal for creating this corpus was to enable (and to begin) progress in research on metalanguage. Between these long-term and immediate goals lies an intermediate step: methods must be developed to detect and delineate metalanguage automatically.

Using the Enhanced Cues Corpus, a two-stage approach to automatic identification of mentioned language is being developed. The first stage is *detection*, the determination of whether a sentence contains an instance of mentioned language. Preliminary results indicate that approximately 70% of instances can be detected using simple machine learning methods (e.g., bag of words input to a decision tree). The remaining instances will require more advanced methods to detect, such as word sense disambiguation to validate occurrences of mention-significant words. The second stage is *delineation*, the determination of the subsequence of words in a sentence that functions as mentioned language. Early efforts have focused on the associations discussed in Section 5 between mentioned language and mention-significant words. The total number of such associations appears to be small, making their collection a tractable activity.

## Acknowledgements

The author would like to thank Don Perlis and Scott Fults for valuable input. This research was supported in part by NSF (under grant #IIS0803739), AFOSR (#FA95500910144), and ONR (#N000140910328).

## References

- Adler, B. Thomas, Luca de Alfaro, Ian Pye & Vishwanath Raman. 2008. Measuring author contributions to the Wikipedia. In *Proc. of WikiSym '08*. New York, NY, USA: ACM.
- American Psychological Association. 2001. *Publication Manual of the American Psychological Association*. 5th ed. Washington, DC: American Psychological Association.
- Anderson, Michael L, Yoshi A Okamoto, Darsana Josyula & Donald Perlis. 2002. The use-mention distinction and its importance to HCI. In *Proc. of EDILOG 2002*. 21–28.
- Anderson, Michael L., Andrew Fister, Bryant Lee & Danny Wang. 2004. On the frequency and types of meta-language in conversation: A preliminary report. In *Proc. of the 14th Annual Conference of the Society for Text & Discourse*.
- Cappelen, H & E Lepore. 1997. Varieties of quotation. *Mind* 106(423). 429–450.
- Chicago Editorial Staff. 2010. *The Chicago Manual of Style*. 16th ed. University of Chicago Press.
- Clark, Herbert H. & Edward F. Schaefer. 1989. Contributing to discourse. *Cognitive Science* 13(2). 259–294.
- Davidson, Donald. 1979. Quotation. *Theory and Decision* 11(1). 27–40.
- Fellbaum, Christiane. 1998. *WordNet: An Electronic Lexical Database*. Cambridge: MIT Press.
- García-Carpintero, Manuel. 2004. The deferred ostension theory of quotation. *Noûs* 38(4). 674–692.
- Klein, Dan & Christopher D. Manning. 2003. Fast exact inference with a factored model for natural language parsing. *Advances in Neural Information Processing Systems* 15.
- Loper, Edward & Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics* 1. 63–70. Association for Computational Linguistics.
- Maier, Emar. 2007. Mixed quotation: Between use and mention. In *Proc. of LENLS 2007*.
- Partee, Barbara. 1973. The syntax and semantics of quotation. In Stephen Anderson & Paul Kiparsky (eds.), *A Festschrift for Morris Halle*. New York: Holt, Rinehart, Winston.
- Perlis, Donald, Khemdut Purang & Carl Andersen. 1998. Conversational adequacy: Mistakes are the essence. *International Journal of Human-Computer Studies* 48(5). 553–575.
- Quine, W. V. O. 1940. *Mathematical Logic*. Cambridge, MA: Harvard University Press.
- Raux, Antoine, Brian Langner, Dan Bohus, Alan W Black & Maxine Eskenazi. 2005. Let's Go public! Taking a spoken dialog system to the real world. In *Proc. of Interspeech 2005*.
- Saka, Paul. 1998. Quotation and the use-mention distinction. *Mind* 107(425). 113–135.
- Saka, Paul. 2003. Quotational constructions. *Belgian Journal of Linguistics* 17(1).
- Strunk, Jr. & E. B. White. 1979. *The Elements of Style, Third Edition*. Macmillan.
- Tarski, Alfred. 1933. The concept of truth in formalized languages. In J. H. Woodger (ed.), *Logic, Semantics, Mathematics*. Oxford: Oxford University Press.
- Wilson, Shomir. 2010. Distinguishing use and mention in natural language. In *Proc. of the NAACL HLT 2010 Student Research Workshop*, 29–33. Association for Computational Linguistics.
- Wilson, Shomir. 2011a. *A Computational Theory of the Use-Mention Distinction in Natural Language*. Ph.D. dissertation, University of Maryland at College Park.
- Wilson, Shomir. 2011b. In search of the use-mention distinction and its impact on language processing tasks. *International Journal of Computational Linguistics and Applications* 2(1-2). 139–154.

# Crosslingual Induction of Semantic Roles

Ivan Titov      Alexandre Klementiev

Saarland University

Saarbrücken, Germany

{titov|aklement}@mmci.uni-saarland.de

## Abstract

We argue that multilingual parallel data provides a valuable source of indirect supervision for induction of shallow semantic representations. Specifically, we consider unsupervised induction of semantic roles from sentences annotated with automatically-predicted syntactic dependency representations and use a state-of-the-art generative Bayesian non-parametric model. At inference time, instead of only seeking the model which explains the monolingual data available for each language, we regularize the objective by introducing a soft constraint penalizing for disagreement in argument labeling on aligned sentences. We propose a simple approximate learning algorithm for our set-up which results in efficient inference. When applied to German-English parallel data, our method obtains a substantial improvement over a model trained without using the agreement signal, when both are tested on non-parallel sentences.

## 1 Introduction

Learning in the context of multiple languages simultaneously has been shown to be beneficial to a number of NLP tasks from morphological analysis to syntactic parsing (Kuhn, 2004; Snyder and Barzilay, 2010; McDonald et al., 2011). The goal of this work is to show that parallel data is useful in unsupervised induction of shallow semantic representations.

Semantic role labeling (SRL) (Gildea and Jurafsky, 2002) involves predicting predicate argument structure, i.e. both the identification of arguments

and their assignment to underlying semantic roles. For example, in the following sentences:

- (a) [<sub>A0</sub>Peter] blamed [<sub>A1</sub>Mary] [<sub>A2</sub>for planning a theft].
- (b) [<sub>A0</sub>Peter] blamed [<sub>A2</sub>planning a theft] [<sub>A1</sub>on Mary].
- (c) [<sub>A1</sub>Mary] was blamed [<sub>A2</sub>for planning a theft] [<sub>A0</sub>by Peter]

the arguments ‘*Peter*’, ‘*Mary*’, and ‘*planning a theft*’ of the predicate ‘*blame*’ take the agent (*A0*), patient (*A1*) and reason (*A2*) roles, respectively. In this work, we focus on predicting argument roles.

SRL representations have many potential applications in NLP and have recently been shown to benefit question answering (Shen and Lapata, 2007; Kaisser and Webber, 2007), textual entailment (Sammons et al., 2009), machine translation (Wu and Fung, 2009; Liu and Gildea, 2010; Wu et al., 2011; Gao and Vogel, 2011), and dialogue systems (Basili et al., 2009; van der Plas et al., 2011), among others. Though syntactic representations are often predictive of semantic roles (Levin, 1993), the interface between syntactic and semantic representations is far from trivial. Lack of simple deterministic rules for mapping syntax to shallow semantics motivates the use of statistical methods.

Most of the current statistical approaches to SRL are supervised, requiring large quantities of human annotated data to estimate model parameters. However, such resources are expensive to create and only available for a small number of languages and domains. Moreover, when moved to a new domain, performance of these models tends to degrade substantially (Pradhan et al., 2008). Sparsity of annotated data motivates the need to look to alternative

resources. In this work, we make use of unsupervised data along with parallel texts and learn to induce semantic structures in two languages simultaneously. As does most of the recent work on unsupervised SRL, we assume that our data is annotated with automatically-predicted syntactic dependency parses and aim to induce a model of linking between syntax and semantics in an unsupervised way.

We expect that both linguistic relatedness and variability can serve to improve semantic parses in individual languages: while the former can provide additional evidence, the latter can serve to reduce uncertainty in ambiguous cases. For example, in our sentences (a) and (b) representing so-called *blame* alternation (Levin, 1993), the same information is conveyed in two different ways and a successful model of semantic role labeling needs to learn the corresponding linkings from the data. Inducing them solely based on monolingual data, though possible, may be tricky as selectional preferences of the roles are not particularly restrictive; similar restrictions for patient and agent roles may further complicate the process. However, both sentences (a) and (b) are likely to be translated in German as ‘ $[_{A_0}Peter]$  *beschuldigte*  $[_{A_1}Mary]$   $[_{A_2}einen Diebstahl zu planen]$ ’. Maximizing agreement between the roles predicted for both languages would provide a strong signal for inducing the proper linkings in our examples.

In this work, we begin with a state-of-the-art monolingual unsupervised Bayesian model (Titov and Klementiev, 2012) and focus on improving its performance in the crosslingual setting. It induces a linking between syntax and semantics, encoded as a clustering of syntactic signatures of predicate arguments. The clustering implicitly defines the set of permissible alternations. For predicates present in both sides of a bitext, we guide models in both languages to prefer clusterings which maximize agreement between predicate argument structures predicted for each aligned predicate pair. We experimentally show the effectiveness of the crosslingual learning on the English-German language pair.

Our model admits efficient inference: the estimation time on CoNLL 2009 data (Hajič et al., 2009) and Europarl v.6 bitext (Koehn, 2005) does not exceed 5 hours on a single processor and the inference algorithm is highly parallelizable, reducing in-

ference time down to less than half an hour on multiple processors. This suggests that the models scale to much larger corpora, which is an important property for a successful unsupervised learning method, as unlabeled data is abundant.

In summary, our contributions are as follows.

- This work is the first to consider the crosslingual setting for unsupervised SRL.
- We propose a form of agreement penalty and show its efficacy on English-German language pair when used in conjunction with a state-of-the-art non-parametric Bayesian model.
- We demonstrate that efficient approximate inference is feasible in the multilingual setting.

The rest of the paper is organized as follows. Section 2 begins with a definition of the crosslingual semantic role induction task we address in this paper. In Section 3, we describe the base monolingual model, and in Section 4 we propose an extension for the crosslingual setting. In Section 5, we describe our inference procedure. Section 6 provides both evaluation and analysis. Finally, additional related work is presented in Section 7.

## 2 Problem Definition

As we mentioned in the introduction, in this work we focus on the labeling stage of semantic role labeling. Identification, though an important problem, can be tackled with heuristics (Lang and Lapata, 2011a; Grenager and Manning, 2006; de Marneffe et al., 2006) or potentially by using a supervised classifier trained on a small amount of data.

Instead of assuming the availability of role annotated data, we rely only on automatically generated syntactic dependency graphs in both languages. While we cannot expect that syntactic structure can trivially map to a semantic representation<sup>1</sup>, we can make use of syntactic cues. In the labeling stage, semantic roles are represented by clusters of arguments, and labeling a particular argument corresponds to deciding on its role cluster. However, instead of dealing with argument occurrences directly,

<sup>1</sup>Although it provides a strong baseline which is difficult to beat (Grenager and Manning, 2006; Lang and Lapata, 2010; Lang and Lapata, 2011a).

we represent them as predicate-specific syntactic signatures, and refer to them as *argument keys*. This representation aids our models in inducing high purity clusters (of argument keys) while reducing their granularity. We follow (Lang and Lapata, 2011a) and use the following syntactic features for English to form the argument key representation:

- Active or passive verb voice (ACT/PASS).
- Arg. position relative to predicate (LEFT/RIGHT).
- Syntactic relation to its governor.
- Preposition used for argument realization.

In the example sentences in Section 1, the argument keys for candidate arguments *Peter* for sentences (a) and (c) would be ACT:LEFT:SBJ and PASS:RIGHT:LGS->by,<sup>2</sup> respectively. While aiming to increase the purity of argument key clusters, this particular representation will not always produce a good match: e.g. *planning a theft* in sentence (b) will have the same key as *Mary* in sentence (a). Increasing the expressiveness of the argument key representation by using features of the syntactic frame would enable us to distinguish that pair of arguments. However, we keep this particular representation, in part to compare with the previous work. In German, we do not include the relative position features, because they are not very informative due to variability in word order.

In sum, we treat the unsupervised semantic role labeling task as clustering of argument keys. Thus, argument occurrences in the corpus whose keys are clustered together are assigned the same semantic role. The objective of this work is to improve argument key clusterings by inducing them simultaneously in two languages.

### 3 Monolingual Model

In this section we describe one of the Bayesian models for semantic role induction proposed in (Titov and Klementiev, 2012). Before describing our method, we briefly introduce the central components of the model: the Chinese Restaurant Processes (CRPs) and Dirichlet Processes (DPs) (Ferguson, 1973; Pitman, 2002). For more details we refer the reader to (Teh, 2007).

<sup>2</sup>LGS denotes a logical subject in a passive construction (Surdeanu et al., 2008).

### 3.1 Chinese Restaurant Processes

CRPs define probability distributions over partitions of a set of objects. An intuitive metaphor for describing CRPs is assignment of tables to restaurant customers. Assume a restaurant with a sequence of tables, and customers who walk into the restaurant one at a time and choose a table to join. The first customer to enter is assigned the first table. Suppose that when a client number  $i$  enters the restaurant,  $i - 1$  customers are sitting at each of the  $k \in (1, \dots, K)$  tables occupied so far. The new customer is then either seated at one of the  $K$  tables with probability  $\frac{N_k}{i-1+\alpha}$ , where  $N_k$  is the number of customers already sitting at table  $k$ , or assigned to a new table with the probability  $\frac{\alpha}{i-1+\alpha}$ ,  $\alpha > 0$ .

If we continue and assume that for each table every customer at a table orders the same meal, with the meal for the table chosen from an arbitrary base distribution  $H$ , then all ordered meals will constitute a sample from the Dirichlet Process  $DP(\alpha, H)$ .

An important property of the non-parametric processes is that a model designer does not need to specify the number of tables (i.e. clusters) a-priori as it is induced automatically on the basis of the data and also depending on the choice of the concentration parameter  $\alpha$ . This property is crucial for our task, as the intended number of roles cannot possibly be specified for every predicate.

### 3.2 The Generative Story

In Section 2 we defined our task as clustering of argument keys, where each cluster corresponds to a semantic role. If an argument key  $k$  is assigned to a role  $r$  ( $k \in r$ ), all of its occurrences are labeled  $r$ .

The Bayesian model encodes two common assumptions about semantic roles. First, it enforces the selectional restriction assumption: namely it stipulates that the distribution over potential argument fillers is sparse for every role, implying that ‘peaky’ distributions of arguments for each role  $r$  are preferred to flat distributions. Second, each role normally appears at most once per predicate occurrence. The inference algorithm will search for a clustering which meets the above requirements to the maximal extent.

The model associates two distributions with each predicate: one governs the selection of argument

fillers for each semantic role, and the other models (and penalizes) duplicate occurrence of roles. Each predicate occurrence is generated independently given these distributions. Let us describe the model by first defining how the set of model parameters and an argument key clustering are drawn, and then explaining the generation of individual predicate and argument instances. The generative story is formally presented in Figure 1.

For each predicate  $p$ , we start by generating a partition of argument keys  $B_p$  with each subset  $r \in B_p$  representing a single semantic role. The partitions are drawn from  $CRP(\alpha)$  independently for each predicate. The crucial part of the model is the set of selectional preference parameters  $\theta_{p,r}$ , the distributions of arguments  $x$  for each role  $r$  of predicate  $p$ . We represent arguments by lemmas of their syntactic heads.<sup>3</sup>

The preference for sparseness of the distributions  $\theta_{p,r}$  is encoded by drawing them from the DP prior  $DP(\beta, H^{(A)})$  with a small concentration parameter  $\beta$ , the base probability distribution  $H^{(A)}$  is just the normalized frequencies of arguments in the corpus. The geometric distribution  $\psi_{p,r}$  is used to model the number of times a role  $r$  appears with a given predicate occurrence. The decision whether to generate at least one role  $r$  is drawn from the uniform Bernoulli distribution. If 0 is drawn then the semantic role is not realized for the given occurrence, otherwise the number of additional roles  $r$  is drawn from the geometric distribution  $Geom(\psi_{p,r})$ . The Beta priors over  $\psi$  can indicate the preference towards generating at most one argument for each role.

Now, when parameters and argument key clusterings are chosen, we can summarize the remainder of the generative story as follows. We begin by independently drawing occurrences for each predicate. For each predicate role we independently decide on the number of role occurrences. Then each of the arguments is generated (see **GenArgument**) by choosing an argument key  $k_{p,r}$  uniformly from the set of argument keys assigned to the cluster  $r$ , and finally choosing its filler  $x_{p,r}$ , where the filler is the lemma of the syntactic head of the argument.

<sup>3</sup>For prepositional phrases, the head noun of the object noun phrase is taken as it encodes crucial lexical information. However, the preposition is not ignored but rather encoded in the corresponding argument key, as explained in Section 2.

Clustering of argument keys:	
for each predicate $p = 1, 2, \dots$ :	
$B_p \sim CRP(\alpha)$	[partition of arg keys]
Parameters:	
for each predicate $p = 1, 2, \dots$ :	
for each role $r \in B_p$ :	
$\theta_{p,r} \sim DP(\beta, H^{(A)})$	[distrib of arg fillers]
$\psi_{p,r} \sim Beta(\eta_0, \eta_1)$	[geom distr for dup roles]
Data generation:	
for each predicate $p = 1, 2, \dots$ :	
for each occurrence $s$ of $p$ :	
for every role $r \in B_p$ :	
if $[n \sim Unif(0, 1)] = 1$ :	[role appears at least once]
<b>GenArgument</b> ( $p, r$ )	[draw one arg]
while $[n \sim \psi_{p,r}] = 1$ :	[continue generation]
<b>GenArgument</b> ( $p, r$ )	[draw more args]
<b>GenArgument</b> ( $p, r$ ):	
$k_{p,r} \sim Unif(1, \dots,  r )$	[draw arg key]
$x_{p,r} \sim \theta_{p,r}$	[draw arg filler]

Figure 1: The generative story for predicate-argument structure.

## 4 Multilingual Extension

As we argued in Section 1, our goal is to penalize for disagreement in semantic structures predicted for each language on parallel data. In doing so, as in much of previous work on unsupervised induction of linguistic structures, we rely on automatically produced word alignments. In Section 6, we describe how we use word alignment to decide if two arguments are aligned; for now, we assume that (noisy) argument alignments are given.

Intuitively, when two arguments are aligned in parallel data, we expect them to be labeled with the same semantic role in both languages. This correspondence is simpler than the one expected in multilingual induction of syntax and morphology where systematic but unknown relation between structures in two language is normally assumed (e.g., (Snyder et al., 2008)). A straightforward implementation of this idea would require us to maintain one-to-one mapping between semantic roles across languages. Instead of assuming this correspondence, we penalize for the lack of isomorphism between the sets of roles in aligned predicates with the penalty dependent on the degree of violation. This softer approach

is more appropriate in our setting, as individual argument keys do not always deterministically map to gold standard roles<sup>4</sup> and strict penalization would result in the propagation of the corresponding over-coarse clusters to the other language. Empirically, we observed this phenomenon on the held-out set with the increase of the penalty weight.

Encoding preference for the isomorphism directly in the generative story is problematic: sparse Dirichlet priors can be used in a fairly trivial way to encode sparsity of the mapping in one direction or another but not in both. Instead, we formalize this preference with a penalty term similar to the expectation criteria in KL-divergence form introduced in McCallum et al. (2007). Specifically, we augment the joint probability with a penalty term computed on parallel data:

$$\sum_{p^{(1)}, p^{(2)}} \left( -\gamma^{(1)} \sum_{r^{(1)} \in B_{p^{(1)}}} f_{r^{(1)}} \arg \max_{r^{(2)} \in B_{p^{(2)}}} \log \hat{P}(r^{(2)} | r^{(1)}) - \gamma^{(2)} \sum_{r^{(2)} \in B_{p^{(2)}}} f_{r^{(2)}} \arg \max_{r^{(1)} \in B_{p^{(1)}}} \log \hat{P}(r^{(1)} | r^{(2)}) \right),$$

where  $\hat{P}(r^{(l)} | r^{(l')})$  is the proportion of times the role  $r^{(l')}$  of predicate  $p^{(l')}$  in language  $l'$  is aligned to the role  $r^{(l)}$  of predicate  $p^{(l)}$  in language  $l$ , and  $f_{r^{(l)}}$  is the total number of times the role is aligned,  $\gamma^{(l)}$  is a non-negative constant. The rationale for introducing the individual weighting  $f_{r^{(l)}}$  is two-fold. First, the proportions  $\hat{P}(r^{(l)} | r^{(l')})$  are more ‘reliable’ when computed from larger counts. Second, more frequent roles should have higher penalty as they compete with the joint probability term, the likelihood part of which scales linearly with role counts.

Space restrictions prevent us from discussing the close relation between this penalty formulation and the existing work on injecting prior and side information in learning objectives in the form of constraints (McCallum et al., 2007; Ganchev et al., 2010; Chang et al., 2007).

In order to support efficient and parallelizable inference, we simplify the above penalty by considering only disjoint pairs of predicates, instead of summing over all pairs  $p^{(1)}$  and  $p^{(2)}$ . When choosing

<sup>4</sup>The average purity for argument keys with automatic argument identification and using predicted syntactic trees, before any clustering, is approximately 90.2% on English and 87.8% on German.

the pairs, we aim to cover the maximal number of alignment counts so as to preserve as much information from parallel corpora as possible. This objective corresponds to the classic maximum weighted bipartite matching problem with the weight for each edge  $p^{(1)}$  and  $p^{(2)}$  equal to the number of times the two predicates were aligned in parallel data. We use the standard polynomial algorithm (the Hungarian algorithm, (Kuhn, 1955)) to find an optimal solution.

## 5 Inference

An inference algorithm for an unsupervised model should be efficient enough to handle vast amounts of unlabeled data, as it can easily be obtained and is likely to improve results. We use a simple approximate inference algorithm based on greedy search. We start by discussing search for the maximum a-posteriori clustering of argument keys in the monolingual set-up and then discuss how it can be extended to accommodate the role alignment penalty.

### 5.1 Monolingual Setting

In the model, a linking between syntax and semantics is induced independently for each predicate. Nevertheless, searching for a MAP clustering can be expensive: even a move involving a single argument key implies some computations for all its occurrences in the corpus. Instead of more complex MAP search algorithms (see, e.g., (Daume III, 2007)), we use a greedy procedure where we start with each argument key assigned to an individual cluster, and then iteratively try to merge clusters. Each move involves (1) choosing an argument key and (2) deciding on a cluster to reassign it to. This is done by considering all clusters (including creating a new one) and choosing the most probable one.

Instead of choosing argument keys randomly at the first stage, we order them by corpus frequency. This ordering is beneficial as getting clustering right for frequent argument keys is more important and the corresponding decisions should be made earlier.<sup>5</sup> We used a single iteration in our experiments, as we have not noticed any benefit from using multiple iterations.

<sup>5</sup>This has been explored before for shallow semantic representations (Lang and Lapata, 2011a; Titov and Klementiev, 2011).

## 5.2 Incorporating the Alignment Penalty

Inference in the monolingual setting is done independently for each predicate, as the model factorizes over the predicates. The role alignment penalty introduces interdependencies between the objectives for each bilingual predicate pair chosen by the assignment algorithm as discussed in Section 4. For each pair of predicates, we search for clusterings to maximize the sum of the log-probability and the negated penalty term.

At first glance it may seem that the alignment penalty can be easily integrated into the greedy MAP search algorithm: instead of considering individual argument keys, one could use pairs of argument keys and decide on their assignment to clusters jointly. However, given that there is no isomorphic mapping between argument keys across languages, this solution is unlikely to be satisfactory.<sup>6</sup> Instead, we use an approximate inference procedure similar in spirit to annotation projection techniques.

For each predicate, we first induce semantic roles independently for the first language, as described in Section 5.1, and then use the same algorithm for the second language but take the penalty term into account. Then we repeat the process in the reverse direction. Among these two solutions, we choose the one which yields the higher objective value. In this way, we begin with producing a clustering for the side which is easier to cluster and provides more clues for the other side.<sup>7</sup>

## 6 Empirical Evaluation

We begin by describing the data and evaluation metrics we use before discussing results.

### 6.1 Data

We run our main experiments on the English-German section of Europarl v6 parallel corpus

---

<sup>6</sup>We also considered a variation of this idea where a pair of argument keys is chosen randomly proportional to their alignment frequency and multiple iterations are repeated. Despite being significantly slower than our method, it did not provide any improvement in accuracy.

<sup>7</sup>In preliminary experiments, we studied an even simpler inference method where the projection direction was fixed for all predicates. Though this approach did outperform the monolingual model, the results were substantially worse than achieved with our method.

(Koehn, 2005) and the CoNLL 2009 distributions of the Penn Treebank WSJ corpus (Marcus et al., 1993) for English and the SALSA corpus (Burchardt et al., 2006) for German. As standard for unsupervised SRL, we use the entire CoNLL training sets for evaluation, and use held-out sets for model selection and parameter tuning.

*Syntactic annotation.* Although the CoNLL 2009 dataset already has predicted dependency structures, we could not reproduce them so that we could use the same parser to annotate Europarl. We chose to reannotate it, since using different parsing models for both datasets would be undesirable. We used MaltParser (Nivre et al., 2007) for English and the syntactic component of the LTH system (Johansson and Nugues, 2008) for German.

*Predicate and argument identification.* We select all non-auxiliary verbs as predicates. For English, we identify their arguments using a heuristic proposed in (Lang and Lapata, 2011a). It is comprised of a list of 8 rules, which use nonlexicalized properties of syntactic paths between a predicate and a candidate argument to iteratively discard non-arguments from the list of all words in a sentence. For German, we use the LTH argument identification classifier. Accuracy of argument identification on CoNLL 2009 using predicted syntactic analyses was 80.7% and 86.5% for English and German, respectively.

*Argument alignment.* We use GIZA++ (Och and Ney, 2003) to produce word alignments in Europarl: we ran it in both directions and kept the intersection of the induced word alignments. For every argument identified in the previous stage, we chose a set of words consisting of the argument’s syntactic head and, for prepositional phrases, the head noun of the object noun phrase. We mark arguments in two languages as aligned if there is any word alignment between the corresponding sets and if they are arguments of aligned predicates.

### 6.2 Evaluation Metrics

We use the standard purity (PU) and collocation (CO) metrics as well as their harmonic mean (F1) to measure the quality of the resulting clusters. Purity measures the degree to which each cluster contains arguments sharing the same gold role:



$$PU = \frac{1}{N} \sum_i \max_j |G_j \cap C_i|$$

where  $C_i$  is the set of arguments in the  $i$ -th induced cluster,  $G_j$  is the set of arguments in the  $j$ th gold cluster, and  $N$  is the total number of arguments. Collocation evaluates the degree to which arguments with the same gold roles are assigned to a single cluster:

$$CO = \frac{1}{N} \sum_j \max_i |G_j \cap C_i|$$

We compute the aggregate PU, CO, and F1 scores over all predicates in the same way as (Lang and Lapata, 2011a) by weighting the scores of each predicate by the number of its argument occurrences. Since our goal is to evaluate the clustering algorithms, we *do not* include incorrectly identified arguments when computing these metrics.

### 6.3 Parameters and Set-up

Our models are robust to parameter settings; the parameters were tuned (to an order of magnitude) to optimize the  $F1$  score on the held-out development set and were as follows. Parameters governing duplicate role generation,  $\eta_0^{(\cdot)}$  and  $\eta_1^{(\cdot)}$ , and penalty weights  $\gamma^{(\cdot)}$  were set to be the same for both languages, and are 100,  $1.e-3$  and 10, respectively. The concentration parameters were set as follows: for English, they were set to  $\alpha^{(1)} = 1.e-3$ ,  $\beta^{(1)} = 1.e-3$ , and, for German, they were  $\alpha^{(2)} = 0.1$ ,  $\beta^{(2)} = 1$ .

Domains of Europarl (parliamentary proceedings) and German/English CoNLL data (newswire) are substantially different. Since the influence of domain shift is not the focus of work, we try to minimize its effect by computing the likelihood part of the objective on CoNLL data alone. This also makes our setting more comparable to prior work.<sup>8</sup>

### 6.4 Results

*Base monolingual model.* We begin by evaluating our base monolingual model *MonoBayes* alone against the current best approaches to unsupervised semantic role induction. Since we do not have access to the systems, we compare on the marginally different English CoNLL 2008 (Surdeanu et al.,

<sup>8</sup>Preliminary experiments on the entire dataset show a slight degradation in performance.

	PU	CO	F1
<i>LLogistic</i>	79.5	76.5	78.0
<i>GraphPart</i>	88.6	70.7	78.6
<i>SplitMerge</i>	88.7	73.0	80.1
<i>MonoBayes</i>	88.1	77.1	<b>82.2</b>
<i>SyntF</i>	81.6	77.5	79.5

Table 1: Argument clustering performance with *gold argument identification* and *gold syntactic parses* on CoNLL 2008 shared-task dataset. Bold-face is used to highlight the best F1 scores.

2008) shared task dataset used in their experiments. We report the results using gold argument identification and gold syntactic parses in order to focus the evaluation on the argument labeling stage and to minimize the noise due to automatic syntactic annotations. The methods are Latent Logistic classification (Lang and Lapata, 2010), Split-Merge clustering (Lang and Lapata, 2011a), and Graph Partitioning (Lang and Lapata, 2011b) (labeled *LLogistic*, *SplitMerge*, and *GraphPart*, respectively) achieving the current best unsupervised SRL results in this setting. Additionally, we compute the syntactic function baseline (*SyntF*), which simply clusters predicate arguments according to the dependency relation to their head. Following (Lang and Lapata, 2010), we allocate a cluster for each of 20 most frequent relations in the CoNLL dataset and one cluster for all other relations. Our model substantially outperforms other models (see Table 1).

*Multilingual extensions.* Next, we improve our model performance using agreement as an additional supervision signal during training (see Section 4). We compare the performance of individual English and German models induced separately (*MonoBayes*) with the jointly induced models (*MultiBayes*) as well as the syntactic baseline, see Table 2.<sup>9</sup> While we see little improvement in F1 for English, the German system improves by 1.8%. For German, the crosslingual learning also results in 1.5% improvement over the syntactic baseline, which is considered difficult to outperform (Grenager and Manning, 2006; Lang and Lapata, 2010). Note that recent unsupervised SRL meth-

<sup>9</sup>Note that the scores are computed on correctly identified arguments only, and tend to be higher in these experiments probably because the complex arguments get discarded by the argument identifier.

	English			German		
	PU	CO	F1	PU	CO	F1
<i>MonoBayes</i>	87.5	80.1	83.6	86.8	75.7	80.9
<i>MultiBayes</i>	86.8	80.7	<b>83.7</b>	85.0	80.6	<b>82.7</b>
<i>SyntF</i>	81.5	79.4	80.4	83.1	79.3	81.2

Table 2: Results on CoNLL 2009 with *automatic argument identification* and *automatic syntactic parses*.

ods do not always improve on it, see Table 1.

The relatively low expressivity and limited purity of our argument keys (see discussion in Section 4) are likely to limit potential improvements when using them in crosslingual learning. The natural next step would be to consider crosslingual learning with a more expressive model of the syntactic frame and syntax-semantics linking.

## 7 Related Work

Unsupervised learning in crosslingual setting has been an active area of research in recent years. However, most of this research has focused on induction of syntactic structures (Kuhn, 2004; Snyder et al., 2009) or morphologic analysis (Snyder and Barzilay, 2008) and we are not aware of any previous work on induction of semantic representations in the crosslingual setting. Learning of semantic representations in the context of monolingual weakly-parallel data was studied in Titov and Kozhevnikov (2010) but their setting was semi-supervised and they experimented only on a restricted domain.

Most of the SRL research has focused on the supervised setting, however, lack of annotated resources for most languages and insufficient coverage provided by the existing resources motivates the need for using unlabeled data or other forms of weak supervision. This includes methods based on graph alignment between labeled and unlabeled data (Fürstenu and Lapata, 2009), using unlabeled data to improve lexical generalization (Deschacht and Moens, 2009), and projection of annotation across languages (Pado and Lapata, 2009; van der Plas et al., 2011). Semi-supervised and weakly-supervised techniques have also been explored for other types of semantic representations but these studies again have mostly focused on restricted domains (Kate and Mooney, 2007; Liang et al., 2009; Goldwasser et al., 2011; Liang et al., 2011).

Early unsupervised approaches to the SRL task include (Swier and Stevenson, 2004), where the VerbNet verb lexicon was used to guide unsupervised learning, and a generative model of Grenager and Manning (2006) which exploits linguistic priors on syntactic-semantic interface.

More recently, the role induction problem has been studied in Lang and Lapata (2010) where it has been reformulated as a problem of detecting alternations and mapping non-standard linkings to the canonical ones. Later, Lang and Lapata (2011a) proposed an algorithmic approach to clustering argument signatures which achieves higher accuracy and outperforms the syntactic baseline. In Lang and Lapata (2011b), the role induction problem is formulated as a graph partitioning problem: each vertex in the graph corresponds to a predicate occurrence and edges represent lexical and syntactic similarities between the occurrences. Unsupervised induction of semantics has also been studied in Poon and Domingos (2009) and Titov and Klementiev (2011) but the induced representations are not entirely compatible with the PropBank-style annotations and they have been evaluated only on a question answering task for the biomedical domain. Also, a related task of unsupervised argument identification has been considered in Abend et al. (2009).

## 8 Conclusions

This work adds unsupervised semantic role labeling to the list of NLP tasks benefiting from the crosslingual induction setting. We show that an agreement signal extracted from parallel data provides indirect supervision capable of substantially improving a state-of-the-art model for semantic role induction.

Although in this work we focused primarily on improving performance for each individual language, cross-lingual semantic representation could be extracted by a simple post-processing step. In future work, we would like to model cross-lingual semantics explicitly.

## Acknowledgements

The work was supported by the MMCI Cluster of Excellence and a Google research award. The authors thank Mikhail Kozhevnikov, Alexis Palmer, Manfred Pinkal, Caroline Sporleder and the anonymous reviewers for their suggestions.

## References

- Omri Abend, Roi Reichart, and Ari Rappoport. 2009. Unsupervised argument identification for semantic role labeling. In *ACL-IJCNLP*.
- Roberto Basili, Diego De Cao, Danilo Croce, Bonaventura Coppola, and Alessandro Moschitti. 2009. Cross-language frame semantics transfer in bilingual corpora. In *CICLING*.
- A. Burchardt, K. Erk, A. Frank, A. Kowalski, S. Pado, and M. Pinkal. 2006. The SALSA corpus: a german corpus resource for lexical semantics. In *LREC*.
- Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *ACL*.
- Hal Daume III. 2007. Fast search for dirichlet process mixture models. In *AISTATS*.
- Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC 2006*.
- Koen Deschacht and Marie-Francine Moens. 2009. Semi-supervised semantic role labeling using the Latent Words Language Model. In *EMNLP*.
- Thomas S. Ferguson. 1973. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230.
- Hagen Fürstenu and Mirella Lapata. 2009. Graph alignment for semi-supervised semantic role labeling. In *EMNLP*.
- Kuzman Ganchev, Joao Graca, Jennifer Gillenwater, and Ben Taskar. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research (JMLR)*, 11:2001–2049.
- Qin Gao and Stephan Vogel. 2011. Corpus expansion for statistical machine translation with semantic role label substitution rules. In *ACL:HLT*.
- Daniel Gildea and Daniel Jurafsky. 2002. Automatic labelling of semantic roles. *Computational Linguistics*, 28(3):245–288.
- Dan Goldwasser, Roi Reichart, James Clarke, and Dan Roth. 2011. Confidence driven unsupervised semantic parsing. In *ACL*.
- Trond Grenager and Christoph Manning. 2006. Unsupervised discovery of a statistical verb lexicon. In *EMNLP*.
- Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *CoNLL 2009: Shared Task*.
- Richard Johansson and Pierre Nugues. 2008. Dependency-based semantic role labeling of Prop-Bank. In *EMNLP*.
- Michael Kaiser and Bonnie Webber. 2007. Question answering based on semantic roles. In *ACL Workshop on Deep Linguistic Processing*.
- Rohit J. Kate and Raymond J. Mooney. 2007. Learning language semantics from ambiguous supervision. In *AAAI*.
- Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the MT Summit*.
- Harold W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.
- Jonas Kuhn. 2004. Experiments in parallel-text based grammar induction. In *ACL*.
- Joel Lang and Mirella Lapata. 2010. Unsupervised induction of semantic roles. In *ACL*.
- Joel Lang and Mirella Lapata. 2011a. Unsupervised semantic role induction via split-merge clustering. In *ACL*.
- Joel Lang and Mirella Lapata. 2011b. Unsupervised semantic role induction with graph partitioning. In *EMNLP*.
- Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press.
- Percy Liang, Michael I. Jordan, and Dan Klein. 2009. Learning semantic correspondences with less supervision. In *ACL-IJCNLP*.
- Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *ACL: HLT*.
- Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Coling*.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.
- Andrew McCallum, Gideon Mann, and Gregory Druck. 2007. Generalized expectation criteria. Technical Report TR 2007-60, University of Massachusetts, Amherst, MA.
- Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *EMNLP*.
- 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 *EMNLP-CoNLL*.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51.

- Sebastian Pado and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36:307–340.
- Jim Pitman. 2002. Poisson-Dirichlet and GEM invariant distributions for split-and-merge transformations of an interval partition. *Combinatorics, Probability and Computing*, 11:501–514.
- Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *EMNLP*.
- Sameer Pradhan, Wayne Ward, and James H. Martin. 2008. Towards robust semantic role labeling. *Computational Linguistics*, 34:289–310.
- M. Sammons, V. Vydiswaran, T. Vieira, N. Johri, M. Chang, D. Goldwasser, V. Srikumar, G. Kundu, Y. Tu, K. Small, J. Rule, Q. Do, and D. Roth. 2009. Relation alignment for textual entailment recognition. In *Text Analysis Conference (TAC)*.
- Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *EMNLP*.
- Benjamin Snyder and Regina Barzilay. 2008. Unsupervised multilingual learning for morphological segmentation. In *ACL*.
- Benjamin Snyder and Regina Barzilay. 2010. Climbing the tower of Babel: Unsupervised multilingual learning. In *ICML*.
- Benjamin Snyder, Tahira Naseem, Jacob Eisenstein, and Regina Barzilay. 2008. Unsupervised multilingual learning for POS tagging. In *EMNLP*.
- Benjamin Snyder, Tahira Naseem, and Regina Barzilay. 2009. Unsupervised multilingual grammar induction. In *ACL*.
- Mihai Surdeanu, Adam Meyers Richard Johansson, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Shared Task*.
- Richard Swier and Suzanne Stevenson. 2004. Unsupervised semantic role labelling. In *EMNLP*.
- Yee Whye Teh. 2007. Dirichlet process. *Encyclopedia of Machine Learning*.
- Ivan Titov and Alexandre Klementiev. 2011. A Bayesian model for unsupervised semantic parsing. In *ACL*.
- Ivan Titov and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *EACL*.
- Ivan Titov and Mikhail Kozhevnikov. 2010. Bootstrapping semantic analyzers from non-contradictory texts. In *ACL*.
- Lonneke van der Plas, Paola Merlo, and James Henderson. 2011. Scaling up automatic cross-lingual semantic role annotation. In *ACL*.
- Dekai Wu and Pascale Fung. 2009. Semantic roles for SMT: A hybrid two-pass model. In *NAACL*.
- Dekai Wu, Marianna Apidianaki, Marine Carpuat, and Lucia Specia, editors. 2011. *Proc. of Fifth Workshop on Syntax, Semantics and Structure in Statistical Translation*. ACL.

# Head-driven Transition-based Parsing with Top-down Prediction

Katsuhiko Hayashi<sup>†</sup>, Taro Watanabe<sup>‡</sup>, Masayuki Asahara<sup>§</sup>, Yuji Matsumoto<sup>†</sup>

<sup>†</sup>Nara Institute of Science and Technology

Ikoma, Nara, 630-0192, Japan

<sup>‡</sup>National Institute of Information and Communications Technology

Sorakugun, Kyoto, 619-0289, Japan

<sup>§</sup>National Institute for Japanese Language and Linguistics

Tachikawa, Tokyo, 190-8561, Japan

katsuhiko-h@is.naist.jp, taro.watanabe@nict.go.jp

masayu-a@ninjal.ac.jp, matsu@is.naist.jp

## Abstract

This paper presents a novel top-down head-driven parsing algorithm for data-driven projective dependency analysis. This algorithm handles global structures, such as clause and coordination, better than shift-reduce or other bottom-up algorithms. Experiments on the English Penn Treebank data and the Chinese CoNLL-06 data show that the proposed algorithm achieves comparable results with other data-driven dependency parsing algorithms.

## 1 Introduction

Transition-based parsing algorithms, such as shift-reduce algorithms (Nivre, 2004; Zhang and Clark, 2008), are widely used for dependency analysis because of the efficiency and comparatively good performance. However, these parsers have one major problem that they can handle only local information. Isozaki et al. (2004) pointed out that the drawbacks of shift-reduce parser could be resolved by incorporating top-down information such as root finding.

This work presents an  $O(n^2)$  top-down head-driven transition-based parsing algorithm which can parse complex structures that are not trivial for shift-reduce parsers. The deductive system is very similar to Earley parsing (Earley, 1970). The Earley prediction is tied to a particular grammar rule, but the proposed algorithm is data-driven, following the current trends of dependency parsing (Nivre, 2006; McDonald and Pereira, 2006; Koo et al., 2010). To do the prediction without any grammar rules, we introduce a weighted prediction that is to predict lower nodes from higher nodes with a statistical model.

To improve parsing flexibility in deterministic parsing, our top-down parser uses beam search algorithm with dynamic programming (Huang and Sagae, 2010). The complexity becomes  $O(n^2 * b)$  where  $b$  is the beam size. To reduce prediction errors, we propose a lookahead technique based on a FIRST function, inspired by the LL(1) parser (Aho and Ullman, 1972). Experimental results show that the proposed top-down parser achieves competitive results with other data-driven parsing algorithms.

## 2 Definition of Dependency Graph

A dependency graph is defined as follows.

**Definition 2.1 (Dependency Graph)** Given an input sentence  $W = \mathbf{n}_0 \dots \mathbf{n}_n$  where  $\mathbf{n}_0$  is a special root node  $\$,$  a directed graph is defined as  $G_W = (V_W, A_W)$  where  $V_W = \{0, 1, \dots, n\}$  is a set of (indices of) nodes and  $A_W \subseteq V_W \times V_W$  is a set of directed arcs. The set of arcs is a set of pairs  $(x, y)$  where  $x$  is a head and  $y$  is a dependent of  $x.$   $x \rightarrow^* l$  denotes a path from  $x$  to  $l.$  A directed graph  $G_W = (V_W, A_W)$  is **well-formed** if and only if:

- There is no node  $x$  such that  $(x, 0) \in A_W.$
- If  $(x, y) \in A_W$  then there is no node  $x'$  such that  $(x', y) \in A_W$  and  $x' \neq x.$
- There is no subset of arcs  $\{(x_0, x_1), (x_1, x_2), \dots, (x_{l-1}, x_l)\} \subseteq A_W$  such that  $x_0 = x_l.$

These conditions are referred to **ROOT, SINGLE-HEAD, and ACYCLICITY,** and we call an **well-formed** directed graph as a **dependency graph.**

**Definition 2.2 (PROJECTIVITY)** A dependency graph  $G_W = (V_W, A_W)$  is **projective** if and only if,

$$\begin{array}{l}
\text{input:} \quad W = \mathbf{n}_0 \dots \mathbf{n}_n \\
\text{axiom}(p_0): \quad 0 : \langle 1, 0, n + 1, \mathbf{n}_0 \rangle : \emptyset \\
\text{pred}_{\curvearrowright}: \quad \frac{\overbrace{\ell : \langle i, h, j, s_d | \dots | s_0 \rangle : -}^{\text{state } p}}{\ell + 1 : \langle i, k, h, s_{d-1} | \dots | s_0 | \mathbf{n}_k \rangle : \{p\}} \exists k : i \leq k < h \\
\text{pred}_{\frown}: \quad \frac{\overbrace{\ell : \langle i, h, j, s_d | \dots | s_0 \rangle : -}^{\text{state } p}}{\ell + 1 : \langle i, k, j, s_{d-1} | \dots | s_0 | \mathbf{n}_k \rangle : \{p\}} \exists k : i \leq k < j \wedge h < i \\
\text{scan:} \quad \frac{\ell : \langle i, h, j, s_d | \dots | s_0 \rangle : \pi}{\ell + 1 : \langle i + 1, h, j, s_d | \dots | s_0 \rangle : \pi} i = h \\
\text{comp:} \quad \frac{\overbrace{- : \langle -, h', j', s'_d | \dots | s'_0 \rangle : \pi'}^{\text{state } q} \quad \overbrace{\ell : \langle i, h, j, s_d | \dots | s_0 \rangle : \pi}^{\text{state } p}}{\ell + 1 : \langle i, h', j', s'_d | \dots | s'_1 | s'_0 \frown s_0 \rangle : \pi'} q \in \pi, h < i \\
\text{goal:} \quad 3n : \langle n + 1, 0, n + 1, s_0 \rangle : \emptyset
\end{array}$$

Figure 1: The non-weighted deductive system of top-down dependency parsing algorithm:  $_-$  means “take anything”.

for every arc  $(x, y) \in A_W$  and node  $l$  in  $x < l < y$  or  $y < l < x$ , there is a path  $x \rightarrow^* l$  or  $y \rightarrow^* l$ .

The proposed algorithm in this paper is for projective dependency graphs. If a projective dependency graph is connected, we call it a **dependency tree**, and if not, a **dependency forest**.

### 3 Top-down Parsing Algorithm

Our proposed algorithm is a transition-based algorithm, which uses stack and queue data structures. This algorithm formally uses the following state:

$$\ell : \langle i, h, j, S \rangle : \pi$$

where  $\ell$  is a step size,  $S$  is a stack of trees  $s_d | \dots | s_0$  where  $s_0$  is a top tree and  $d$  is a window size for feature extraction,  $i$  is an index of node on the top of the input node queue,  $h$  is an index of root node of  $s_0$ ,  $j$  is an index to indicate the right limit ( $j - 1$  inclusive) of  $\text{pred}_{\curvearrowright}$ , and  $\pi$  is a set of pointers to **predictor states**, which are states just before putting the node in  $h$  onto stack  $S$ . In the deterministic case,  $\pi$  is a singleton set except for the initial state.

This algorithm has four actions,  $\text{predict}_{\curvearrowright}(\text{pred}_{\curvearrowright})$ ,  $\text{predict}_{\frown}(\text{pred}_{\frown})$ , scan and complete(comp). The deductive system of the top-down algorithm is shown in Figure 1. The initial state  $p_0$  is a state initialized by an artificial root node  $\mathbf{n}_0$ . This algorithm

applies one action to each state selected from applicable actions in each step. Each of three kinds of actions, pred, scan, and comp, occurs  $n$  times, and this system takes  $3n$  steps for a complete analysis.

Action  $\text{pred}_{\curvearrowright}$  puts  $\mathbf{n}_k$  onto stack  $S$  selected from the input queue in the range,  $i \leq k < h$ , which is to the left of the root  $\mathbf{n}_h$  in the stack top. Similarly, action  $\text{pred}_{\frown}$  puts a node  $\mathbf{n}_k$  onto stack  $S$  selected from the input queue in the range,  $h < i \leq k < j$ , which is to the right of the root  $\mathbf{n}_h$  in the stack top. The node  $\mathbf{n}_i$  on the top of the queue is scanned if it is equal to the root node  $\mathbf{n}_h$  in the stack top. Action comp creates a directed arc  $(h', h)$  from the root  $h'$  of  $s'_0$  on a predictor state  $q$  to the root  $h$  of  $s_0$  on a current state  $p$  if  $h < i$ <sup>1</sup>.

The precondition  $i < h$  of action  $\text{pred}_{\curvearrowright}$  means that the input nodes in  $i \leq k < h$  have not been predicted yet.  $\text{Pred}_{\curvearrowright}$ , scan and  $\text{pred}_{\frown}$  do not conflict with each other since their preconditions  $i < h$ ,  $i = h$  and  $h < i$  do not hold at the same time. However, this algorithm faces a  $\text{pred}_{\frown}$ -comp conflict because both actions share the same precondition  $h < i$ , which means that the input nodes in  $1 \leq k \leq h$  have been predicted and scanned. This

<sup>1</sup>In a single root tree, the special root symbol  $\$_0$  has exactly one child node. Therefore, we do not apply comp action to a state if its condition satisfies  $s_1 \cdot \mathbf{h} = \mathbf{n}_0 \wedge \ell \neq 3n - 1$ .

step	state	stack	queue	action	state information
0	$p_0$	$\$0$	$I_1 \text{ saw}_2 \text{ a}_3 \text{ girl}_4$	–	$\langle 1, 0, 5 \rangle : \emptyset$
1	$p_1$	$\$0   \text{saw}_2$	$I_1 \text{ saw}_2 \text{ a}_3 \text{ girl}_4$	$\text{pred}_{\curvearrowright}$	$\langle 1, 2, 5 \rangle : \{p_0\}$
2	$p_2$	$\text{saw}_2   I_1$	$I_1 \text{ saw}_2 \text{ a}_3 \text{ girl}_4$	$\text{pred}_{\curvearrowright}$	$\langle 1, 1, 2 \rangle : \{p_1\}$
3	$p_3$	$\text{saw}_2   I_1$	$\text{saw}_2 \text{ a}_3 \text{ girl}_4$	scan	$\langle 2, 1, 2 \rangle : \{p_1\}$
4	$p_4$	$\$0   I_1 \frown \text{saw}_2$	$\text{saw}_2 \text{ a}_3 \text{ girl}_4$	comp	$\langle 2, 2, 5 \rangle : \{p_0\}$
5	$p_5$	$\$0   I_1 \frown \text{saw}_2$	$\text{a}_3 \text{ girl}_4$	scan	$\langle 3, 2, 5 \rangle : \{p_0\}$
6	$p_6$	$I_1 \frown \text{saw}_2   \text{girl}_4$	$\text{a}_3 \text{ girl}_4$	$\text{pred}_{\curvearrowright}$	$\langle 3, 4, 5 \rangle : \{p_5\}$
7	$p_7$	$\text{girl}_4   \text{a}_3$	$\text{a}_3 \text{ girl}_4$	$\text{pred}_{\curvearrowright}$	$\langle 3, 3, 4 \rangle : \{p_6\}$
8	$p_8$	$\text{girl}_4   \text{a}_3$	$\text{girl}_4$	scan	$\langle 4, 3, 4 \rangle : \{p_6\}$
9	$p_9$	$I_1 \frown \text{saw}_2   \text{a}_3 \frown \text{girl}_4$	$\text{girl}_4$	comp	$\langle 4, 4, 5 \rangle : \{p_5\}$
10	$p_{10}$	$I_1 \frown \text{saw}_2   \text{a}_3 \frown \text{girl}_4$		scan	$\langle 5, 4, 5 \rangle : \{p_5\}$
11	$p_{11}$	$\$0   I_1 \frown \text{saw}_2 \frown \text{girl}_4$		comp	$\langle 5, 2, 5 \rangle : \{p_0\}$
12	$p_{12}$	$\$0 \frown \text{saw}_2$		comp	$\langle 5, 0, 5 \rangle : \emptyset$

Figure 2: Stages of the top-down deterministic parsing process for a sentence “I saw a girl”. We follow a convention and write the stack with its topmost element to the right, and the queue with its first element to the left. In this example, we set the window size  $d$  to 1, and write the descendants of trees on stack elements  $s_0$  and  $s_1$  within depth 1.

parser constructs left and right children of a head node in a left-to-right direction by scanning the head node prior to its right children. Figure 2 shows an example for parsing a sentence “I saw a girl”.

#### 4 Correctness

To prove the *correctness* of the system in Figure 1 for the **projective** dependency graph, we use the proof strategy of (Nivre, 2008a). The correct deductive system is both *sound* and *complete*.

**Theorem 4.1** *The deductive system in Figure 1 is correct for the class of dependency forest.*

**Proof 4.1** *To show soundness, we show that  $G_{p_0} = (V_W, \emptyset)$ , which is a directed graph defined by the axiom, is well-formed and projective, and that every transition preserves this property.*

- **ROOT:** *The node 0 is a root in  $G_{p_0}$ , and the node 0 is on the top of stack of  $p_0$ . The two  $\text{pred}$  actions put a word onto the top of stack, and predict an arc from root or its descendant to the child. The  $\text{comp}$  actions add the predicted arcs which include no arc of  $(x, 0)$ .*
- **SINGLE-HEAD:**  *$G_{p_0}$  is single-head. A node  $y$  is no longer in stack and queue after a  $\text{comp}$  action creates an arc  $(x, y)$ . The node  $y$  cannot make any arc  $(x', y)$  after the removal.*
- **ACYCLICITY:**  *$G_{p_0}$  is acyclic. A cycle is created only if an arc  $(x, y)$  is added when there is a directed path  $y \rightarrow^* x$ . The node  $x$  is no*

*longer in stack and queue when the directed path  $y \rightarrow^* x$  was made by adding an arc  $(l, x)$ . There is no chance to add the arc  $(x, y)$  on the directed path  $y \rightarrow^* x$ .*

- **PROJECTIVITY:**  *$G_{p_0}$  is projective. Projectivity is violated by adding an arc  $(x, y)$  when there is a node  $l$  in  $x < l < y$  or  $y < l < x$  with the path to or from the outside of the span  $x$  and  $y$ . When  $\text{pred}_{\curvearrowright}$  creates an arc relation from  $x$  to  $y$ , the node  $y$  cannot be scanned before all nodes  $l$  in  $x < l < y$  are scanned and completed. When  $\text{pred}_{\curvearrowright}$  creates an arc relation from  $x$  to  $y$ , the node  $y$  cannot be scanned before all nodes  $k$  in  $k < y$  are scanned and completed, and the node  $x$  cannot be scanned before all nodes  $l$  in  $y < l < x$  are scanned and completed. In those processes, the node  $l$  in  $x < l < y$  or  $y < l < x$  does not make a path to or from the outside of the span  $x$  and  $y$ , and a path  $x \rightarrow^* l$  or  $y \rightarrow^* l$  is created.  $\square$*

*To show completeness, we show that for any sentence  $W$ , and dependency forest  $G_W = (V_W, A_W)$ , there is a transition sequence  $C_{0,m}$  such that  $G_{p_m} = G_W$  by an inductive method.*

- *If  $|W| = 1$ , the projective dependency graph for  $W$  is  $G_W = (\{0\}, \emptyset)$  and  $G_{p_0} = G_W$ .*
- *Assume that the claim holds for sentences with length less or equal to  $t$ , and assume that  $|W| = t + 1$  and  $G_W = (V_W, A_W)$ . The subgraph  $G_{W'}$  is defined as  $(V_W - t, A^{-t})$  where*

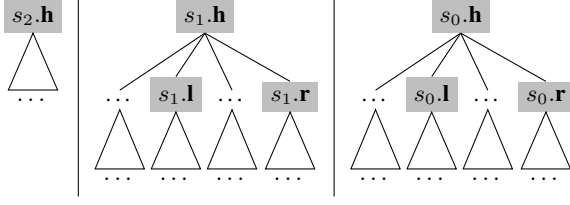


Figure 3: Feature window of trees on stack  $S$ : The window size  $d$  is set to 2. Each  $x.h$ ,  $x.l$  and  $x.r$  denotes root, left and right child nodes of a stack element  $x$ .

$A^{-t} = A_W - \{(x, y) | x = t \vee y = t\}$ . If  $G_W$  is a dependency forest, then  $G_{W'}$  is also a dependency forest. It is obvious that there is a transition sequence for constructing  $G_W$  except arcs which have a node  $t$  as a head or a dependent<sup>2</sup>. There is a state  $p_q = q : \langle i, x, t + 1 \rangle : -$  for  $i$  and  $x$  ( $0 \leq x < i < t + 1$ ). When  $x$  is the head of  $t$ ,  $\text{pred}_{\curvearrowright}$  to  $t$  creates a state  $p_{q+1} = q + 1 : \langle i, t, t + 1 \rangle : \{p_q\}$ . At least one node  $y$  in  $i \leq y < t$  becomes the dependent of  $t$  by  $\text{pred}_{\curvearrowright}$  and there is a transition sequence for constructing a tree rooted by  $y$ . After constructing a subtree rooted by  $t$  and spanned from  $i$  to  $t$ ,  $t$  is scanned, and then  $\text{comp}$  creates an arc from  $x$  to  $t$ . It is obvious that the remaining transition sequence exists. Therefore, we can construct a transition sequence  $C_{0,m}$  such that  $G_{p_m} = G_W$ .  $\square$

The deductive system in Figure 1 is both sound and complete. Therefore, it is correct.  $\square$

## 5 Weighted Parsing Model

### 5.1 Stack-based Model

The proposed algorithm employs a stack-based model for scoring hypothesis. The cost of the model is defined as follows:

$$c_s(i, h, j, S) = \theta_s \cdot \mathbf{f}_{s,act}(i, h, j, S) \quad (1)$$

where  $\theta_s$  is a weight vector,  $\mathbf{f}_s$  is a feature function, and  $act$  is one of the applicable actions to a state  $\ell : \langle i, h, j, S \rangle : \pi$ . We use a set of feature templates of (Huang and Sagae, 2010) for the model. As shown in Figure 3, left children  $s_0.l$  and  $s_1.l$  of trees on

<sup>2</sup>This transition sequence is defined for  $G_{W'}$ , but it is possible to be regarded as the definition for  $G_W$  as long as the transition sequence is indifferent from the node  $t$ .

---

### Algorithm 1 Top-down Parsing with Beam Search

---

```

1: input  $W = \mathbf{n}_0, \dots, \mathbf{n}_n$ 
2:  $start \leftarrow \langle 1, 0, n + 1, \mathbf{n}_0 \rangle$ 
3:  $buf[0] \leftarrow \{start\}$ 
4: for  $\ell \leftarrow 1 \dots 3n$  do
5:    $hypo \leftarrow \{\}$ 
6:   for each  $state$  in  $buf[\ell - 1]$  do
7:     for  $act \leftarrow \text{applicableAct}(state)$  do
8:        $newstates \leftarrow \text{actor}(act, state)$ 
9:       addAll  $newstates$  to  $hypo$ 
10:  add top  $b$  states to  $buf[\ell]$  from  $hypo$ 
11: return best candidate from  $buf[3n]$ 

```

---

stack for extracting features are different from those of Huang and Sagae (2010) because in our parser the left children are generated from left to right.

As mentioned in Section 1, we apply beam search and Huang and Sagae (2010)’s DP techniques to our top-down parser. Algorithm 1 shows the our beam search algorithm in which top most  $b$  states are preserved in a buffer  $buf[\ell]$  in each step. In line 10 of Algorithm 1, equivalent states in the step  $\ell$  are merged following the idea of DP. Two states  $\langle i, h, j, S \rangle$  and  $\langle i', h', j', S' \rangle$  in the step  $\ell$  are equivalent, notated  $\langle i, h, j, S \rangle \sim \langle i', h', j', S' \rangle$ , iff

$$\mathbf{f}_{s,act}(i, h, j, S) = \mathbf{f}_{s,act}(i', h', j', S'). \quad (2)$$

When two equivalent predicted states are merged, their predictor states in  $\pi$  get combined. For further details about this technique, readers may refer to (Huang and Sagae, 2010).

### 5.2 Weighted Prediction

The step 0 in Figure 2 shows an example of prediction for a head node “\$0”, where the node “saw<sub>2</sub>” is selected as its child node. To select a probable child node, we define a statistical model for the prediction. In this paper, we integrate the cost from a graph-based model (McDonald and Pereira, 2006) which directly models dependency links. The cost of the 1st-order model is defined as the relation between a child node  $\mathbf{c}$  and a head node  $\mathbf{h}$ :

$$c_p(\mathbf{h}, \mathbf{c}) = \theta_p \cdot \mathbf{f}_p(\mathbf{h}, \mathbf{c}) \quad (3)$$

where  $\theta_p$  is a weight vector and  $\mathbf{f}_p$  is a features function. Using the cost  $c_p$ , the top-down parser selects a probable child node in each prediction step.

When we apply beam search to the top-down parser, then we no longer use  $\exists$  but  $\forall$  on  $\text{pred}_{\curvearrowright}$  and



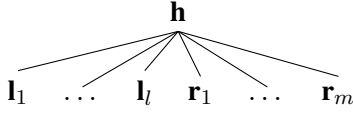


Figure 4: An example of tree structure: Each  $\mathbf{h}$ ,  $\mathbf{l}_l$  and  $\mathbf{r}_1$  denotes head, left and right child nodes.

$\text{pred}_{\curvearrowright}$  in Figure 1. Therefore, the parser may predict many nodes as an appropriate child from a single state, causing many predicted states. This may cause the beam buffer to be filled only with the states, and these may exclude other states, such as scanned or completed states. Thus, we limit the number of predicted states from a single state by **prediction size** implicitly in line 10 of Algorithm 1.

To improve the prediction accuracy, we introduce a more sophisticated model. The cost of the sibling 2nd-order model is defined as the relationship between  $\mathbf{c}$ ,  $\mathbf{h}$  and a sibling node **sib**:

$$c_p(\mathbf{h}, \mathbf{sib}, \mathbf{c}) = \theta_p \cdot \mathbf{f}_p(\mathbf{h}, \mathbf{sib}, \mathbf{c}). \quad (4)$$

The 1st- and sibling 2nd-order models are the same as McDonald and Pereira (2006)'s definitions, except the cost factors of the sibling 2nd-order model. The cost factors for a tree structure in Figure 4 are defined as follows:

$$c_p(\mathbf{h}, -, \mathbf{l}_1) + \sum_{y=1}^{l-1} c_p(\mathbf{h}, \mathbf{l}_y, \mathbf{l}_{y+1}) \\ + c_p(\mathbf{h}, -, \mathbf{r}_1) + \sum_{y=1}^{m-1} c_p(\mathbf{h}, \mathbf{r}_y, \mathbf{r}_{y+1}).$$

This is different from McDonald and Pereira (2006) in that the cost factors for left children are calculated from left to right, while those in McDonald and Pereira (2006)'s definition are calculated from right to left. This is because our top-down parser generates left children from left to right. Note that the cost of weighted prediction model in this section is incrementally calculated by using only the information on the current state, thus the condition of state merge in Equation 2 remains unchanged.

### 5.3 Weighted Deductive System

We extend deductive system to a weighted one, and introduce **forward cost** and **inside cost** (Stolcke,

1995; Huang and Sagae, 2010). The forward cost is the total cost of a sequence from an initial state to the end state. The inside cost is the cost of a top tree  $s_0$  in stack  $S$ . We define these costs using a combination of stack-based model and weighted prediction model. The forward and inside costs of the combination model are as follows:

$$\begin{cases} c^{\text{fw}} = c_s^{\text{fw}} + c_p^{\text{fw}} \\ c^{\text{in}} = c_s^{\text{in}} + c_p^{\text{in}} \end{cases} \quad (5)$$

where  $c_s^{\text{fw}}$  and  $c_s^{\text{in}}$  are a forward cost and an inside cost for stack-based model, and  $c_p^{\text{fw}}$  and  $c_p^{\text{in}}$  are a forward cost and an inside cost for weighted prediction model. We add the following tuple of costs to a state:

$$(c_s^{\text{fw}}, c_s^{\text{in}}, c_p^{\text{fw}}, c_p^{\text{in}}).$$

For each action, we define how to efficiently calculate the forward and inside costs<sup>3</sup>, following Stolcke (1995) and Huang and Sagae (2010)'s works. In either case of  $\text{pred}_{\curvearrowleft}$  or  $\text{pred}_{\curvearrowright}$ ,

$$\frac{(c_s^{\text{fw}}, -, c_p^{\text{fw}}, -)}{(c_s^{\text{fw}} + \lambda, 0, c_p^{\text{fw}} + c_p(s_0 \cdot \mathbf{h}, \mathbf{n}_k), 0)}$$

where

$$\lambda = \begin{cases} \theta_s \cdot \mathbf{f}_{s, \text{pred}_{\curvearrowleft}}(i, h, j, S) & \text{if } \text{pred}_{\curvearrowleft} \\ \theta_s \cdot \mathbf{f}_{s, \text{pred}_{\curvearrowright}}(i, h, j, S) & \text{if } \text{pred}_{\curvearrowright} \end{cases} \quad (6)$$

In the case of scan,

$$\frac{(c_s^{\text{fw}}, c_s^{\text{in}}, c_p^{\text{fw}}, c_p^{\text{in}})}{(c_s^{\text{fw}} + \xi, c_s^{\text{in}} + \xi, c_p^{\text{fw}}, c_p^{\text{in}})}$$

where

$$\xi = \theta_s \cdot \mathbf{f}_{s, \text{scan}}(i, h, j, S). \quad (7)$$

In the case of comp,

$$\frac{(c_s^{\text{fw}}, c_s^{\text{in}}, c_p^{\text{fw}}, c_p^{\text{in}})}{(c_s^{\text{fw}} + c_s^{\text{in}} + \mu, c_s^{\text{in}} + c_s^{\text{in}} + \mu, \\ c_p^{\text{fw}} + c_p^{\text{in}} + c_p(s'_0 \cdot \mathbf{h}, s_0 \cdot \mathbf{h}), \\ c_p^{\text{in}} + c_p^{\text{in}} + c_p(s'_0 \cdot \mathbf{h}, s_0 \cdot \mathbf{h}))}$$

where

$$\mu = \theta_s \cdot \mathbf{f}_{s, \text{comp}}(i, h, j, S) + \theta_s \cdot \mathbf{f}_{s, \text{pred}_{\curvearrowleft}}(-, h', j', S'). \quad (8)$$

<sup>3</sup>For brevity, we present the formula not by 2nd-order model as equation 4 but a 1st-order one for weighted prediction.

Pred<sub>∧</sub> takes either pred<sub>∧</sub> or pred<sub>∨</sub>. Beam search is performed based on the following linear order for the two states  $p$  and  $p'$  at the same step, which have  $(c^{\text{fw}}, c^{\text{in}})$  and  $(c'^{\text{fw}}, c'^{\text{in}})$  respectively:

$$p \succ p' \text{ iff } c^{\text{fw}} < c'^{\text{fw}} \text{ or } c^{\text{fw}} = c'^{\text{fw}} \wedge c^{\text{in}} < c'^{\text{in}}. \quad (9)$$

We prioritize the forward cost over the inside cost since forward cost pertains to longer action sequence and is better suited to evaluate hypothesis states than inside cost (Nederhof, 2003).

#### 5.4 FIRST Function for Lookahead

Top-down backtrack parser usually reduces backtracking by precomputing the set FIRST( $\cdot$ ) (Aho and Ullman, 1972). We define the set FIRST( $\cdot$ ) for our top-down dependency parser:

$$\text{FIRST}(t') = \{\text{ld.t} \mid \text{ld} \in \text{Imdescendant}(\text{Tree}, t') \\ \text{Tree} \in \text{Corpus}\} \quad (10)$$

where  $t'$  is a POS-tag,  $\text{Tree}$  is a correct dependency tree which exists in  $\text{Corpus}$ , a function  $\text{Imdescendant}(\text{Tree}, t')$  returns the set of the leftmost descendant node  $\text{ld}$  of each nodes in  $\text{Tree}$  whose POS-tag is  $t'$ , and  $\text{ld.t}$  denotes a POS-tag of  $\text{ld}$ . Though our parser does not backtrack, it looks ahead when selecting possible child nodes at the prediction step by using the function FIRST. In case of pred<sub>∧</sub>:

$$\forall k : i \leq k < h \wedge \mathbf{n}_i.t \in \text{FIRST}(\mathbf{n}_k.t) \\ \frac{\overbrace{\ell : \langle i, h, j, s_d | \dots | s_0 \rangle : -}^{\text{state } p}}{\ell + 1 : \langle i, k, h, s_{d-1} | \dots | s_0 | \mathbf{n}_k \rangle : \{p\}}$$

where  $\mathbf{n}_i.t$  is a POS-tag of the node  $\mathbf{n}_i$  on the top of the queue, and  $\mathbf{n}_k.t$  is a POS-tag in  $k$ th position of an input nodes. The case for pred<sub>∨</sub> is the same. If there are no nodes which satisfy the condition, our top-down parser creates new states for all nodes, and pushes them into *hypo* in line 9 of Algorithm 1.

#### 6 Time Complexity

Our proposed top-down algorithm has three kinds of actions which are scan, comp and predict. Each scan and comp actions occurs  $n$  times when parsing a sentence with the length  $n$ . Predict action also occurs  $n$  times in which a child node is selected from

a node sequence in the input queue. Thus, the algorithm takes the following times for prediction:

$$n + (n - 1) + \dots + 1 = \sum_i^n i = \frac{n(n + 1)}{2}. \quad (11)$$

As  $n^2$  for prediction is the most dominant factor, the time complexity of the algorithm is  $O(n^2)$  and that of the algorithm with beam search is  $O(n^2 * b)$ .

#### 7 Related Work

Alshawi (1996) proposed head automaton which recognizes an input sentence top-down. Eisner and Satta (1999) showed that there is a cubic-time parsing algorithm on the formalism of the head automaton grammars, which are equivalently converted into split-head bilexical context-free grammars (SBCFGs) (McAllester, 1999; Johnson, 2007). Although our proposed algorithm does not employ the formalism of SBCFGs, it creates left children before right children, implying that it does not have spurious ambiguities as well as parsing algorithms on the SBCFGs. Head-corner parsing algorithm (Kay, 1989) creates dependency tree top-down, and in this our algorithm has similar spirit to it.

Yamada and Matsumoto (2003) applied a shift-reduce algorithm to dependency analysis, which is known as arc-standard transition-based algorithm (Nivre, 2004). Nivre (2003) proposed another transition-based algorithm, known as arc-eager algorithm. The arc-eager algorithm processes right-dependent top-down, but this does not involve the prediction of lower nodes from higher nodes. Therefore, the arc-eager algorithm is a totally bottom-up algorithm. Zhang and Clark (2008) proposed a combination approach of the transition-based algorithm with graph-based algorithm (McDonald and Pereira, 2006), which is the same as our combination model of stack-based and prediction models.

#### 8 Experiments

Experiments were performed on the English Penn Treebank data and the Chinese CoNLL-06 data. For the English data, we split WSJ part of it into sections 02-21 for training, section 22 for development and section 23 for testing. We used Yamada and Matsumoto (2003)'s head rules to convert phrase structure to dependency structure. For the Chinese data,

	time	accuracy	complete	root
McDonald05,06 (2nd)	0.15	90.9, 91.5	37.5, 42.1	–
Koo10 (Koo and Collins, 2010)	–	93.04	–	–
Hayashi11 (Hayashi et al., 2011)	0.3	92.89	–	–
2nd-MST*	0.13	92.3	43.7	96.0
Goldberg10 (Goldberg and Elhadad, 2010)	–	89.7	37.5	91.5
Kitagawa10 (Kitagawa and Tanaka-Ishii, 2010)	–	91.3	41.7	–
Zhang08 (Sh beam 64)	–	91.4	41.8	–
Zhang08 (Sh+Graph beam 64)	–	92.1	45.4	–
Huang10 (beam+DP)	0.04	92.1	–	–
Huang10* (beam 8, 16, 32+DP)	0.03, 0.06, 0.10	92.3, 92.27, 92.26	43.5, 43.7, 43.8	96.0, 96.0, 96.1
Zhang11 (beam 64) (Zhang and Nivre, 2011)	–	93.07	49.59	–
top-down* (beam 8, 16, 32+pred 5+DP)	0.07, 0.12, 0.22	91.7, 92.3, 92.5	45.0, 45.7, <b>45.9</b>	94.5, 95.7, 96.2
top-down* (beam 8, 16, 32+pred 5+DP+FIRST)	0.07, 0.12, 0.22	91.9, 92.4, <b>92.6</b>	45.0, 45.3, 45.5	95.1, 96.2, <b>96.6</b>

Table 1: Results for test data: Time measures the parsing time per sentence in seconds. Accuracy is an unlabeled attachment score, complete is a sentence complete rate, and root is a correct root rate. \* indicates our experiments.

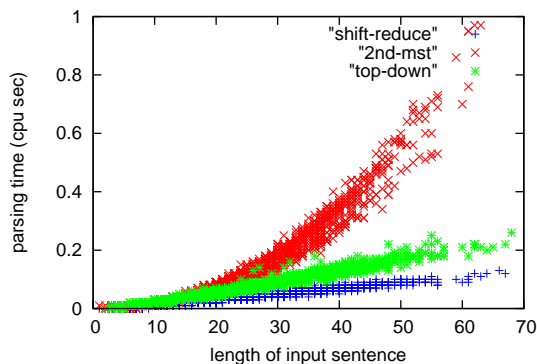


Figure 5: Scatter plot of parsing time against sentence length, comparing with top-down, 2nd-MST and shift-reduce parsers (beam size: 8, pred size: 5)

we used the information of words and fine-grained POS-tags for features. We also implemented and experimented Huang and Sagae (2010)’s arc-standard shift-reduce parser. For the 2nd-order Eisner-Satta algorithm, we used MSTParser (McDonald, 2012).

We used an early update version of averaged perceptron algorithm (Collins and Roark, 2004) for training of shift-reduce and top-down parsers. A set of feature templates in (Huang and Sagae, 2010) were used for the stack-based model, and a set of feature templates in (McDonald and Pereira, 2006) were used for the 2nd-order prediction model. The weighted prediction and stack-based models of top-down parser were jointly trained.

### 8.1 Results for English Data

During training, we fixed the prediction size and beam size to 5 and 16, respectively, judged by pre-

	accuracy	complete	root
oracle (sh+mst)	94.3	52.3	97.7
oracle (top+sh)	94.2	51.7	97.6
oracle (top+mst)	93.8	50.7	97.1
oracle (top+sh+mst)	94.9	55.3	98.1

Table 2: Oracle score, choosing the highest accuracy parse for each sentence on test data from results of top-down (beam 8, pred 5) and shift-reduce (beam 8) and MST(2nd) parsers in Table 1.

	accuracy	complete	root
top-down (beam:8, pred:5)	90.9	80.4	93.0
shift-reduce (beam:8)	90.8	77.6	93.5
2nd-MST	91.4	79.3	94.2
oracle (sh+mst)	94.0	85.1	95.9
oracle (top+sh)	93.8	84.0	95.6
oracle (top+mst)	93.6	84.2	95.3
oracle (top+sh+mst)	94.7	86.5	96.3

Table 3: Results for Chinese Data (CoNLL-06)

liminary experiments on development data. After 25 iterations of perceptron training, we achieved 92.94 unlabeled accuracy for top-down parser with the FIRST function and 93.01 unlabeled accuracy for shift-reduce parser on development data by setting the beam size to 8 for both parsers and the prediction size to 5 in top-down parser. These trained models were used for the following testing.

We compared top-down parsing algorithm with other data-driven parsing algorithms in Table 1. Top-down parser achieved comparable unlabeled accuracy with others, and outperformed them on the sentence complete rate. On the other hand, top-down parser was less accurate than shift-reduce

No.717	Little <u>Lily</u> , as Ms. Cunningham calls <sub>7</sub> herself in the book , really										was <sub>14</sub>	n't ordinary .										
shift-reduce	2	<u>7</u>	2	2	6	4	14	7	7	11	9	7	14	0	14	14	14					
2nd-MST	2	<u>14</u>	2	2	6	7	4	7	7	11	9	2	14	0	14	14	14					
top-down	2	<u>14</u>	2	2	6	7	4	7	7	11	9	2	14	0	14	14	14					
correct	2	<u>14</u>	2	2	6	7	4	7	7	11	9	2	14	0	14	14	14					
No.127	resin , used to make garbage bags , milk jugs , housewares , toys and meat										packaging <sub>25</sub>	, among other items .										
shift-reduce	25	9	9	13	11	15	<u>13</u>	<u>25</u>	18	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	7	<u>25</u>	<u>25</u>	29	27	4
2nd-MST	29	9	9	13	11	15	<u>13</u>	<u>29</u>	18	<u>29</u>	<u>29</u>	<u>29</u>	<u>29</u>	<u>25</u>	<u>25</u>	<u>25</u>	29	<u>25</u>	<u>25</u>	29	7	4
top-down	7	9	9	13	11	15	<u>25</u>	<u>25</u>	18	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	13	<u>25</u>	<u>25</u>	29	27	4
correct	7	9	9	13	11	15	<u>25</u>	<u>25</u>	18	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	<u>25</u>	13	<u>25</u>	<u>25</u>	29	27	4

Table 4: Two examples on which top-down parser is superior to two bottom-up parsers: In correct analysis, the boxed portion is the head of the underlined portion. Bottom-up parsers often mistake to capture the relation.

parser on the correct root measure. In step 0, top-down parser predicts a child node, a root node of a complete tree, using little syntactic information, which may lead to errors in the root node selection. Therefore, we think that it is important to seek more suitable features for the prediction in future work.

Figure 5 presents the parsing time against sentence length. Our proposed top-down parser is theoretically slower than shift-reduce parser and Figure 5 empirically indicates the trends. The dominant factor comes from the score calculation, and we will leave it for future work. Table 2 shows the oracle score for test data, which is the score of the highest accuracy parse selected for each sentence from results of several parsers. This indicates that the parses produced by each parser are different from each other. However, the gains obtained by the combination of top-down and 2nd-MST parsers are smaller than other combinations. This is because top-down parser uses the same features as 2nd-MST parser, and these are more effective than those of stack-based model. It is worth noting that as shown in Figure 5, our  $O(n^2 * b)$  ( $b = 8$ ) top-down parser is much faster than  $O(n^3)$  Eisner-Satta CKY parsing.

## 8.2 Results for Chinese Data (CoNLL-06)

We also experimented on the Chinese data. Following English experiments, shift-reduce parser was trained by setting beam size to 16, and top-down parser was trained with the beam size and the prediction size to 16 and 5, respectively. Table 3 shows the results on the Chinese test data when setting beam size to 8 for both parsers and prediction size to 5 in top-down parser. The trends of the results are almost

the same as those of the English results.

## 8.3 Analysis of Results

Table 4 shows two interesting results, on which top-down parser is superior to either shift-reduce parser or 2nd-MST parser. The sentence No.717 contains an adverbial clause structure between the subject and the main verb. Top-down parser is able to handle the long-distance dependency while shift-reduce parser cannot correctly analyze it. The effectiveness on the clause structures implies that our head-driven parser may handle non-projective structures well, which are introduced by Johansson’s head rule (Johansson and Nugues, 2007). The sentence No.127 contains a coordination structure, which it is difficult for bottom-up parsers to handle, but, top-down parser handles it well because its top-down prediction globally captures the coordination.

## 9 Conclusion

This paper presents a novel head-driven parsing algorithm and empirically shows that it is as practical as other dependency parsing algorithms. Our head-driven parser has potential for handling non-projective structures better than other non-projective dependency algorithms (McDonald et al., 2005; Attardi, 2006; Nivre, 2008b; Koo et al., 2010). We are in the process of extending our head-driven parser for non-projective structures as our future work.

## Acknowledgments

We would like to thank Kevin Duh for his helpful comments and to the anonymous reviewers for giving valuable comments.

## References

- A. V. Aho and J. D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1: Parsing. Prentice-Hall.
- H. Alshawi. 1996. Head automata for speech translation. In *Proc. the ICSLP*.
- G. Attardi. 2006. Experiments with a multilanguage non-projective dependency parser. In *Proc. the 10th CoNLL*, pages 166–170.
- M. Collins and B. Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proc. the 42nd ACL*.
- J. Earley. 1970. An efficient context-free parsing algorithm. *Communications of the Association for Computing Machinery*, 13(2):94–102.
- J. M. Eisner and G. Satta. 1999. Efficient parsing for bilexical context-free grammars and head automaton grammars. In *Proc. the 37th ACL*, pages 457–464.
- Y. Goldberg and M. Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. In *Proc. the HLT-NAACL*, pages 742–750.
- K. Hayashi, T. Watanabe, M. Asahara, and Y. Matsumoto. 2011. The third-order variational reranking on packed-shared dependency forests. In *Proc. EMNLP*, pages 1479–1488.
- L. Huang and K. Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proc. the 48th ACL*, pages 1077–1086.
- H. Isozaki, H. Kazawa, and T. Hirao. 2004. A deterministic word dependency analyzer enhanced with preference learning. In *Proc. the 21st COLING*, pages 275–281.
- R. Johansson and P. Nugues. 2007. Extended constituent-to-dependency conversion for english. In *Proc. NODALIDA*.
- M. Johnson. 2007. Transforming projective bilexical dependency grammars into efficiently-parsable CFGs with unfold-fold. In *Proc. the 45th ACL*, pages 168–175.
- M. Kay. 1989. Head driven parsing. In *Proc. the IWPT*.
- K. Kitagawa and K. Tanaka-Ishii. 2010. Tree-based deterministic dependency parsing — an application to nivre’s method —. In *Proc. the 48th ACL 2010 Short Papers*, pages 189–193, July.
- T. Koo and M. Collins. 2010. Efficient third-order dependency parsers. In *Proc. the 48th ACL*, pages 1–11.
- T. Koo, A. M. Rush, M. Collins, T. Jaakkola, and D. Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proc. EMNLP*, pages 1288–1298.
- D. McAllester. 1999. A reformulation of eisner and satta’s cubic time parser for split head automata grammars. <http://ttic.uchicago.edu/dmcallester/>.
- R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proc. EACL*, pages 81–88.
- R. McDonald, F. Pereira, K. Ribarov, and J. Hajic. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proc. HLT-EMNLP*, pages 523–530.
- R. McDonald. 2012. Minimum spanning tree parser. <http://www.seas.upenn.edu/strctlrn/MSTParser>.
- M.-J. Nederhof. 2003. Weighted deductive parsing and knuth’s algorithm. *Computational Linguistics*, 29:135–143.
- J. Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proc. the IWPT*, pages 149–160.
- J. Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proc. the ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57.
- J. Nivre. 2006. *Inductive Dependency Parsing*. Springer.
- J. Nivre. 2008a. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34:513–553.
- J. Nivre. 2008b. Sorting out dependency parsing. In *Proc. the CoTAL*, pages 16–27.
- A. Stolcke. 1995. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 21(2):165–201.
- H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proc. the IWPT*, pages 195–206.
- Y. Zhang and S. Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proc. EMNLP*, pages 562–571.
- Y. Zhang and J. Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proc. the 49th ACL*, pages 188–193.

# MIX Is Not a Tree-Adjoining Language

**Makoto Kanazawa**

National Institute of Informatics  
2-1-2 Hitotsubashi, Chiyoda-ku  
Tokyo, 101-8430, Japan  
kanazawa@nii.ac.jp

**Sylvain Salvati**

INRIA Bordeaux Sud-Ouest, LaBRI  
351, Cours de la Libération  
F-33405 Talence Cedex, France  
sylvain.salvati@labri.fr

## Abstract

The language MIX consists of all strings over the three-letter alphabet  $\{a, b, c\}$  that contain an equal number of occurrences of each letter. We prove Joshi's (1985) conjecture that MIX is not a tree-adjoining language.

## 1 Introduction

The language

$$\text{MIX} = \{ w \in \{a, b, c\}^* \mid |w|_a = |w|_b = |w|_c \}$$

has attracted considerable attention in computational linguistics.<sup>1</sup> This language was used by Bach (1981) in an exercise to show that the permutation closure of a context-free language is not necessarily context-free.<sup>2</sup> MIX may be considered a prototypical example of *free word order language*, but, as remarked by Bach (1981), it seems that no human language “has such complete freedom for order”, because “typically, certain constituents act as ‘boundary domains’ for scrambling”. Joshi (1985) refers to MIX as representing “an extreme case of the degree of free word order permitted in a language”, which is “linguistically not relevant”. Gazdar (1988) adopts a similar position regarding the relation between MIX

<sup>1</sup>If  $w$  is a string and  $d$  is a symbol, we write  $|w|_d$  to mean the number of occurrences of  $d$  in  $w$ . We will use the notation  $|w|$  to denote the length of  $w$ , i.e., the total number of occurrences of symbols in  $w$ .

<sup>2</sup>According to Gazdar (1988), “MIX was originally described by Emmon Bach and was so-dubbed by students in the 1983 Hampshire College Summer Studies in Mathematics”. According to Bach (1988), the name MIX was “the happy invention of Bill Marsh”.

and natural languages, noting that “it seems rather unlikely that any natural language will turn out to have a MIX-like characteristic”.

It therefore seems natural to assume that languages such as MIX should be excluded from any class of formal languages that purports to be a tight formal characterization of the possible natural languages. It was in this spirit that Joshi et al. (1991) suggested that MIX should not be in the class of so-called *mildly context-sensitive* languages:

“[mildly context-sensitive grammars] capture only certain kinds of dependencies, e.g., nested dependencies and certain limited kinds of cross-serial dependencies (for example, in the subordinate clause constructions in Dutch or some variations of them, but perhaps not in the so-called MIX (or Bach) language) . . .”

Mild context-sensitivity is an informally defined notion first introduced by Joshi (1985); it consists of the three conditions of *limited cross-serial dependencies*, *constant growth*, and *polynomial parsing*. The first condition is only vaguely formulated, but the other two conditions are clearly satisfied by tree-adjoining grammars. The suggestion of Joshi et al. (1991) was that MIX should be regarded as a violation of the condition of limited cross-serial dependencies.

Joshi (1985) conjectured rather strongly that MIX is not a tree-adjoining language: “TAGs cannot generate this language, although for TAGs the proof is not in hand yet”. An even stronger conjecture was made by Marsh (1985), namely, that MIX is not an

*indexed language*.<sup>3</sup> (It is known that the indexed languages properly include the tree-adjoining languages.) Joshi et al. (1991), however, expressed a more pessimistic view about the conjecture:

“It is not known whether TAG ... can generate MIX. This has turned out to be a very difficult problem. In fact, it is not even known whether an IG [(indexed grammar)] can generate MIX.”

This open question has become all the more pressing after a recent result by Salvati (2011). This result says that MIX is in the class of *multiple context-free languages* (Seki et al., 1991), or equivalently, languages of *linear context-free rewriting systems* (Vijay-Shanker et al., 1987; Weir, 1988), which has been customarily regarded as a formal counterpart of the informal notion of a mildly context-sensitive language.<sup>4</sup> It means that either we have to abandon the identification of multiple context-free languages with mildly context-sensitive languages, or we should revise our conception of limited cross-serial dependencies and stop regarding MIX-like languages as violations of this condition. Surely, the resolution of Joshi’s (1985) conjecture should crucially affect the choice between these two alternatives.

In this paper, we prove that MIX is not a tree-adjoining language. Our proof is cast in terms of the formalism of *head grammar* (Pollard, 1984; Roach, 1987), which is known to be equivalent to TAG (Vijay-Shanker and Weir, 1994). The key to our proof is the notion of an *n-decomposition* of a string over  $\{a, b, c\}$ , which is similar to the notion of a derivation in head grammars, but independent of any particular grammar. The parameter  $n$  indicates how unbalanced the occurrence counts of the three letters can be at any point in a decomposition. We first

<sup>3</sup>The relation of MIX with indexed languages is also of interest in combinatorial group theory. Gilman (2005) remarks that “it does not ... seem to be known whether or not the word problem of  $Z \times Z$  is indexed”, alluding to the language  $O_2 = \{w \in \{a, \bar{a}, b, \bar{b}\}^* \mid |w|_a = |w|_{\bar{a}}, |w|_b = |w|_{\bar{b}}\}$ . Since  $O_2$  and MIX are rationally equivalent,  $O_2$  is indexed if and only if MIX is indexed (Salvati, 2011).

<sup>4</sup>Joshi et al. (1991) presented linear context-free rewriting systems as mildly context-sensitive grammars. Groenink (1997) wrote “The class of mildly context-sensitive languages seems to be most adequately approached by LCFRS.”

show that if MIX is generated by some head grammar, then there is an  $n$  such that every string in MIX has an  $n$ -decomposition. We then prove that if every string in MIX has an  $n$ -decomposition, then every string in MIX must have a 2-decomposition. Finally, we exhibit a particular string in MIX that has no 2-decomposition. The length of this string is 87, and the fact that it has no 2-decomposition was first verified by a computer program accompanying this paper. We include here a rigorous, mathematical proof of this fact not relying on the computer verification.

## 2 Head Grammars

A *head grammar* is a quadruple  $G = (N, \Sigma, P, S)$ , where  $N$  is a finite set of nonterminals,  $\Sigma$  is a finite set of terminal symbols (alphabet),  $S$  is a distinguished element of  $N$ , and  $P$  is a finite set of rules. Each nonterminal is interpreted as a binary predicate on strings in  $\Sigma^*$ . There are four types of rules:

$$\begin{aligned} A(x_1 x_2 y_1, y_2) &\leftarrow B(x_1, x_2), C(y_1, y_2) \\ A(x_1, x_2 y_1 y_2) &\leftarrow B(x_1, x_2), C(y_1, y_2) \\ A(x_1 y_1, y_2 x_2) &\leftarrow B(x_1, x_2), C(y_1, y_2) \\ A(w_1, w_2) &\leftarrow \end{aligned}$$

Here,  $A, B, C \in N$ ,  $x_1, x_2, y_1, y_2$  are *variables*, and  $w_1, w_2 \in \Sigma \cup \{\varepsilon\}$ .<sup>5</sup> Rules of the first three types are *binary rules* and rules of the last type are *terminating rules*. This definition of a head grammar actually corresponds to a normal form for head grammars that appears in section 3.3 of Vijay-Shanker and Weir’s (1994) paper.<sup>6</sup>

The rules of head grammars are interpreted as implications from right to left, where variables can be instantiated to any terminal strings. Each binary

<sup>5</sup>We use  $\varepsilon$  to denote the empty string.

<sup>6</sup>This normal form is also mentioned in chapter 5, section 4 of Kracht’s (2003) book. The notation we use to express rules of head grammars is borrowed from *elementary formal systems* (Smullyan, 1961; Arikawa et al., 1992), also known as *literal movement grammars* (Groenink, 1997; Kracht, 2003), which are logic programs over strings. In Vijay-Shanker and Weir’s (1994) notation, the four rules are expressed as follows:

$$\begin{aligned} A &\rightarrow C_{2,2}(B, C) \\ A &\rightarrow C_{1,2}(B, C) \\ A &\rightarrow W(B, C) \\ A &\rightarrow C_{1,1}(w_1 \uparrow w_2) \end{aligned}$$

rule involves an operation that combines two pairs of strings to form a new pair. The operation involved in the third rule is known as *wrapping*; the operations involved in the first two rules we call *left concatenation* and *right concatenation*, respectively. If  $G = (N, \Sigma, P, S)$  is a head grammar,  $A \in N$ , and  $w_1, w_2 \in \Sigma^*$ , then we say that a *fact*  $A(w_1, w_2)$  is *derivable* and write  $\vdash_G A(w_1, w_2)$ , if  $A(w_1, w_2)$  can be inferred using the rules in  $P$ . More formally, we have  $\vdash_G A(w_1, w_2)$  if one of the following conditions holds:

- $A(w_1, w_2) \leftarrow$  is a terminating rule in  $P$ .
- $\vdash_G B(u_1, u_2)$ ,  $\vdash_G C(v_1, v_2)$ , and there is a binary rule  $A(\alpha_1, \alpha_2) \leftarrow B(x_1, x_2), C(y_1, y_2)$  in  $P$  such that  $(w_1, w_2)$  is the result of substituting  $u_1, u_2, v_1, v_2$  for  $x_1, x_2, y_1, y_2$ , respectively, in  $(\alpha_1, \alpha_2)$ .

The language of  $G$  is

$$L(G) = \{ w_1 w_2 \mid \vdash_G S(w_1, w_2) \}.$$

**Example 1.** Let  $G = (N, \Sigma, P, S)$ , where  $N = \{S, A, A', C, D, E, F\}$ ,  $\Sigma = \{a, \bar{a}, \#\}$ , and  $P$  consists of the following rules:

$$\begin{aligned} S(x_1 y_1, y_2 x_2) &\leftarrow D(x_1, x_2), C(y_1, y_2) \\ C(\varepsilon, \#) &\leftarrow \\ D(\varepsilon, \varepsilon) &\leftarrow \\ D(x_1 y_1, y_2 x_2) &\leftarrow F(x_1, x_2), D(y_1, y_2) \\ F(x_1 y_1, y_2 x_2) &\leftarrow A(x_1, x_2), E(y_1, y_2) \\ A(a, a) &\leftarrow \\ E(x_1 y_1, y_2 x_2) &\leftarrow D(x_1, x_2), A'(y_1, y_2) \\ A'(\bar{a}, \bar{a}) &\leftarrow \end{aligned}$$

We have  $L(G) = \{ w \# w^R \mid w \in D_{\{a, \bar{a}\}} \}$ , where  $D_{\{a, \bar{a}\}}$  is the Dyck language over  $\{a, \bar{a}\}$  and  $w^R$  is the reversal of  $w$ . All binary rules of this grammar are wrapping rules.

If  $\vdash_G A(w_1, w_2)$ , a *derivation tree* for  $A(w_1, w_2)$  is a finite binary tree whose nodes are labeled by facts that are derived during the derivation of  $A(w_1, w_2)$ . A derivation tree for  $A(w_1, w_2)$  represents a “proof” of  $\vdash_G A(w_1, w_2)$ , and is formally defined as follows:

- If  $A(w_1, w_2) \leftarrow$  is a terminating rule, then a tree with a single node labeled by  $A(w_1, w_2)$  is a derivation tree for  $A(w_1, w_2)$ .

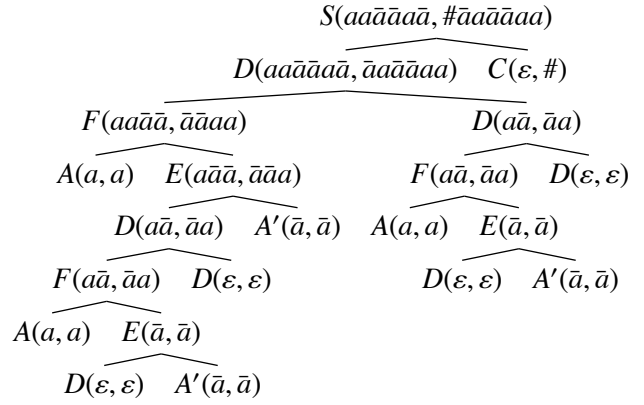


Figure 1: An example of a derivation tree of a head grammar.

- If  $\vdash_G A(w_1, w_2)$  is derived from  $\vdash_G B(u_1, u_2)$  and  $\vdash_G C(v_1, v_2)$  by some binary rule, then a binary tree whose root is labeled by  $A(w_1, w_2)$  and whose immediate left (right) subtree is a derivation tree for  $B(u_1, u_2)$  (for  $C(v_1, v_2)$ , respectively) is a derivation tree for  $A(w_1, w_2)$ .

If  $w \in L(G)$ , a derivation tree for  $w$  is a derivation tree for some  $S(w_1, w_2)$  such that  $w_1 w_2 = w$ .

**Example 1 (continued).** Figure 1 shows a derivation tree for  $aaāāāā\#āāāāāā$ .

The following lemma should be intuitively clear from the definition of a derivation tree:

**Lemma 1.** Let  $G = (N, \Sigma, P, S)$  be a head grammar and  $A$  be a nonterminal in  $N$ . Suppose that  $w \in L(G)$  has a derivation tree in which a fact  $A(v_1, v_2)$  appears as a label of a node. Then there are strings  $z_0, z_1, z_2$  with the following properties:

- $w = z_0 v_1 z_1 v_2 z_2$ , and
- $\vdash_G A(u_1, u_2)$  implies  $z_0 u_1 z_1 u_2 z_2 \in L(G)$ .

*Proof.* We can prove by straightforward induction on the height of derivation trees that whenever  $A(v_1, v_2)$  appears on a node in a derivation tree for  $B(w_1, w_2)$ , then there exist  $z_0, z_1, z_2, z_3$  that satisfy one of the following conditions:

- $w_1 = z_0 v_1 z_1 v_2 z_2$ ,  $w_2 = z_3$ , and  $\vdash_G A(u_1, u_2)$  implies  $\vdash_G B(z_0 u_1 z_1 u_2 z_2, z_3)$ .
- $w_1 = z_0$ ,  $w_2 = z_1 v_1 z_2 v_2 z_3$ , and  $\vdash_G A(u_1, u_2)$  implies  $\vdash_G B(z_0, z_1 u_1 z_2 u_2 z_3)$ .



(c)  $w_1 = z_0v_1z_1$ ,  $w_2 = z_2v_2z_3$ , and  $\vdash_G A(u_1, u_2)$  implies  $\vdash_G B(z_0u_1z_1, z_2u_2z_3)$ .

We omit the details.  $\square$

We call a nonterminal  $A$  of a head grammar  $G$  *useless* if  $A$  does not appear in any derivation trees for strings in  $L(G)$ . Clearly, useless nonterminals can be eliminated from any head grammar without affecting the language of the grammar.

### 3 Decompositions of Strings in MIX

Henceforth,  $\Sigma = \{a, b, c\}$ . Let  $\mathbb{Z}$  denote the set of integers. Define functions  $\psi_1, \psi_2: \Sigma^* \rightarrow \mathbb{Z}$ ,  $\psi: \Sigma^* \rightarrow \mathbb{Z} \times \mathbb{Z}$  by

$$\begin{aligned}\psi_1(w) &= |w|_a - |w|_c, \\ \psi_2(w) &= |w|_b - |w|_c, \\ \psi(w) &= (\psi_1(w), \psi_2(w)).\end{aligned}$$

Clearly, we have  $\psi(a) = (1, 0)$ ,  $\psi(b) = (0, 1)$ ,  $\psi(c) = (-1, -1)$ , and

$$w \in \text{MIX} \quad \text{iff} \quad \psi(w) = (0, 0).$$

Note that for all strings  $w_1, w_2 \in \Sigma^*$ ,  $\psi(w_1w_2) = \psi(w_1) + \psi(w_2)$ . In other words,  $\psi$  is a homomorphism from the free monoid  $\Sigma^*$  to  $\mathbb{Z} \times \mathbb{Z}$  with addition as the monoid operation and  $(0, 0)$  as identity.

**Lemma 2.** *Suppose that  $G = (N, \Sigma, P, S)$  is a head grammar without useless nonterminals such that  $L(G) \subseteq \text{MIX}$ . There exists a function  $\Psi_G: N \rightarrow \mathbb{Z} \times \mathbb{Z}$  such that  $\vdash_G A(u_1, u_2)$  implies  $\psi(u_1u_2) = \Psi_G(A)$ .*

*Proof.* Since  $G$  has no useless nonterminals, for each nonterminal  $A$  of  $G$ , there is a derivation tree for some string in  $L(G)$  in which  $A$  appears in a node label. By Lemma 1, there are strings  $z_0, z_1, z_2$  such that  $\vdash_G A(u_1, u_2)$  implies  $z_0u_1z_1u_2z_2 \in L(G)$ . Since  $L(G) \subseteq \text{MIX}$ , we have  $\psi(z_0u_1z_1u_2z_2) = (0, 0)$ , and hence

$$\psi(u_1u_2) = -\psi(z_0z_1z_2). \quad \square$$

A *decomposition* of  $w \in \Sigma^*$  is a finite binary tree satisfying the following conditions:

- the root is labeled by some  $(w_1, w_2)$  such that  $w = w_1w_2$ ,

- each internal node whose left and right children are labeled by  $(u_1, u_2)$  and  $(v_1, v_2)$ , respectively, is labeled by one of  $(u_1u_2v_1, v_2)$ ,  $(u_1, u_2v_1v_2)$ ,  $(u_1v_1, v_2u_2)$ .
- each leaf node is labeled by some  $(s_1, s_2)$  such that  $s_1s_2 \in \{b, c\}^* \cup \{a, c\}^* \cup \{a, b\}^*$ .

Thus, the label of an internal node in a decomposition is obtained from the labels of its children by left concatenation, right concatenation, or wrapping. It is easy to see that if  $G$  is a head grammar over the alphabet  $\Sigma$ , any derivation for  $w \in L(G)$  induces a decomposition of  $w$ . (Just strip off nonterminals.) Note that unlike with derivation trees, we have placed no bound on the length of a string that may appear on a leaf node of a decomposition. This will be convenient in some of the proofs below.

When  $p$  and  $q$  are integers, we write  $[p, q]$  for the set  $\{r \in \mathbb{Z} \mid p \leq r \leq q\}$ . We call a decomposition of  $w$  an *n-decomposition* if each of its nodes is labeled by some  $(v_1, v_2)$  such that  $\psi(v_1v_2) \in [-n, n] \times [-n, n]$ .

**Lemma 3.** *If  $\text{MIX} = L(G)$  for some head grammar  $G = (\Sigma, N, P, S)$ , then there exists an  $n$  such that each  $w \in \text{MIX}$  has an  $n$ -decomposition.*

*Proof.* We may suppose without loss of generality that  $G$  has no useless nonterminal. Since  $\text{MIX} = L(G)$ , there is a function  $\Psi_G$  satisfying the condition of Lemma 2. Since the set  $N$  of nonterminals of  $G$  is finite, there is an  $n$  such that  $\Psi_G(A) \in [-n, n] \times [-n, n]$  for all  $A \in N$ . Then it is clear that a derivation tree for  $w \in L(G)$  induces an  $n$ -decomposition of  $w$ .  $\square$

If  $w = d_1 \dots d_m \in \Sigma^m$ , then for  $0 \leq i \leq j \leq m$ , we write  $w[i, j]$  to refer to the substring  $d_{i+1} \dots d_j$  of  $w$ . (As a special case, we have  $w[i, i] = \varepsilon$ .) The following is a key lemma in our proof:

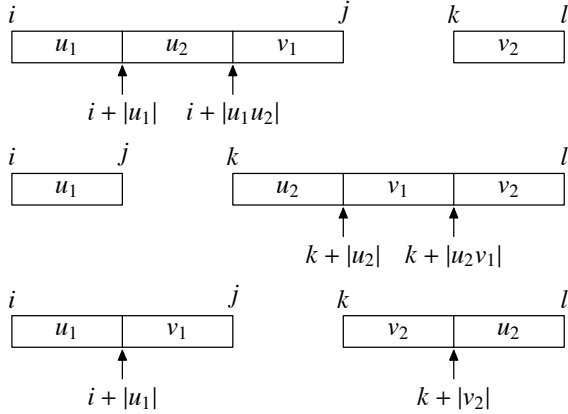
**Lemma 4.** *If each  $w \in \text{MIX}$  has an  $n$ -decomposition, then each  $w \in \text{MIX}$  has a 2-decomposition.*

*Proof.* Assume that each  $w \in \text{MIX}$  has an  $n$ -decomposition. Define a homomorphism  $\gamma_n: \Sigma^* \rightarrow \Sigma^*$  by

$$\begin{aligned}\gamma_n(a) &= a^n, \\ \gamma_n(b) &= b^n, \\ \gamma_n(c) &= c^n.\end{aligned}$$

Clearly,  $\gamma_n$  is an injection, and we have  $\psi(\gamma_n(v)) = n \cdot \psi(v)$  for all  $v \in \Sigma^*$ .

Let  $w \in \text{MIX}$  with  $|w| = m$ . Then  $w' = \gamma_n(w) \in \text{MIX}$  and  $|w'| = mn$ . By assumption,  $w'$  has an  $n$ -decomposition  $\mathcal{D}$ . We assign a 4-tuple  $(i, j, k, l)$  of natural numbers to each node of  $\mathcal{D}$  in such a way that  $(w'[i, j], w'[k, l])$  equals the label of the node. This is done recursively in an obvious way, starting from the root. If the root is labeled by  $(w_1, w_2)$ , then it is assigned  $(0, |w_1|, |w_1|, |w_1 w_2|)$ . If a node is assigned a tuple  $(i, j, k, l)$  and has two children labeled by  $(u_1, u_2)$  and  $(v_1, v_2)$ , respectively, then the 4-tuples assigned to the children are determined according to how  $(u_1, u_2)$  and  $(v_1, v_2)$  are combined at the parent node:



Now define a function  $f: [0, mn] \rightarrow \{kn \mid 0 \leq k \leq m\}$  by

$$f(i) = \begin{cases} i & \text{if } n \text{ divides } i, \\ n \cdot \lfloor i/n \rfloor & \text{if } n \text{ does not divide } i \text{ and} \\ & w'[i-1, i] \in \{a, b\}, \\ n \cdot \lceil i/n \rceil & \text{if } n \text{ does not divide } i \text{ and} \\ & w'[i-1, i] = c. \end{cases}$$

Clearly,  $f$  is weakly increasing in the sense that  $i \leq j$  implies  $f(i) \leq f(j)$ . Let  $\mathcal{D}'$  be the result of replacing the label of each node in  $\mathcal{D}$  by

$$(w'[f(i), f(j)], w'[f(k), f(l)]),$$

where  $(i, j, k, l)$  is the 4-tuple of natural numbers assigned to that node by the above procedure. It is easy to see that  $\mathcal{D}'$  is another decomposition of  $w'$ . Note that since each of  $f(i), f(j), f(k), f(l)$  is an integral multiple of  $n$ , we always have

$$(w'[f(i), f(j)], w'[f(k), f(l)]) = (\gamma_n(u), \gamma_n(v))$$

for some substrings  $u, v$  of  $w$ . This implies that for  $h = 1, 2$ ,

$$\psi_h(w'[f(i), f(j)]w'[f(k), f(l)])$$

is an integral multiple of  $n$ .

*Claim.*  $\mathcal{D}'$  is a  $2n$ -decomposition.

We have to show that every node label  $(v_1, v_2)$  in  $\mathcal{D}'$  satisfies  $\psi(v_1 v_2) \in [-2n, 2n] \times [-2n, 2n]$ . For  $h = 1, 2$ , define  $\varphi_h: [0, mn] \times [0, mn] \rightarrow \mathbb{Z}$  as follows:

$$\varphi_h(i, j) = \begin{cases} \psi_h(w'[i, j]) & \text{if } i \leq j, \\ -\psi_h(w'[j, i]) & \text{otherwise.} \end{cases}$$

Then it is easy to see that for all  $i, j, i', j' \in [0, mn]$ ,

$$\varphi_h(i', j') = \varphi_h(i', i) + \varphi_h(i, j) + \varphi_h(j, j').$$

Inspecting the definition of the function  $f$ , we can check that

$$\varphi_h(f(i), i) \in [0, n-1]$$

always holds. Suppose that  $(i, j, k, l)$  is assigned to a node in  $\mathcal{D}$ . By assumption, we have  $\psi_h(w'[i, j]w'[k, l]) \in [-n, n]$ , and

$$\begin{aligned} & \psi_h(w'[f(i), f(j)]w'[f(k), f(l)]) \\ &= \psi_h(w'[f(i), f(j)]) + \psi_h(w'[f(k), f(l)]) \\ &= \varphi_h(f(i), f(j)) + \varphi_h(f(k), f(l)) \\ &= \varphi_h(f(i), i) + \varphi_h(i, j) + \varphi_h(j, f(j)) \\ & \quad + \varphi_h(f(k), k) + \varphi_h(k, l) + \varphi_h(l, f(l)) \\ &= \varphi_h(f(i), i) + \psi_h(w'[i, j]) + \varphi_h(j, f(j)) \\ & \quad + \varphi_h(f(k), k) + \psi_h(w'[k, l]) + \varphi_h(l, f(l)) \\ &= \psi_h(w'[i, j]w'[k, l]) + \varphi_h(f(i), i) + \varphi_h(f(k), k) \\ & \quad + \varphi_h(j, f(j)) + \varphi_h(l, f(l)) \\ &\in \{p + q_1 + q_2 + r_1 + r_2 \mid p \in [-n, n], \\ & \quad q_1, q_2 \in [0, n-1], r_1, r_2 \in [-n+1, 0]\} \\ &= [-3n+2, 3n-2]. \end{aligned}$$

Since  $\psi_h(w'[f(i), f(j)]w'[f(k), f(l)])$  must be an integral multiple of  $n$ , it follows that

$$\psi_h(w'[f(i), f(j)]w'[f(k), f(l)]) \in \{-2n, -n, 0, n, 2n\}.$$

This establishes the claim.

We have shown that each node of  $\mathcal{D}'$  is labeled by a pair of strings of the form  $(\gamma_n(u), \gamma_n(v))$  such that

$$\psi(\gamma_n(u)\gamma_n(v)) \in \{-2n, -n, 0, n, 2n\} \times \{-2n, -n, 0, n, 2n\}.$$

Now it is easy to see that inverting the homomorphism  $\gamma_n$  at each node of  $\mathcal{D}'$

$$(\gamma_n(u), \gamma_n(v)) \mapsto (u, v)$$

gives a 2-decomposition of  $w$ . □

#### 4 A String in MIX That Has No 2-Decomposition

By Lemmas 3 and 4, in order to prove that there is no head grammar for MIX, it suffices to exhibit a string in MIX that has no 2-decomposition. The following is such a string:

$$z = a^5 b^{14} a^{19} c^{29} b^{15} a^5.$$

In this section, we prove that the string  $z$  has no 2-decomposition.<sup>7</sup>

It helps to visualize strings in MIX as closed curves in a plane. If  $w$  is a string in MIX, by plotting the coordinates of  $\psi(v)$  for each prefix  $v$  of  $w$ , we can represent  $w$  by a closed curve  $C$  together with a map  $t: [0, |w|] \rightarrow C$ . The representation of the string  $z$  is given in Figure 2.

Let us call a string  $w \in \{a, b, c\}^*$  such that  $\psi(w) \in [-2, 2] \times [-2, 2]$  *long* if  $w$  contains all three letters, and *short* otherwise. (If  $\psi(w) \notin [-2, 2] \times [-2, 2]$ , then  $w$  is neither short nor long.) It is easy to see that a short string  $w$  always satisfies

$$|w|_a \leq 4, \quad |w|_b \leq 4, \quad |w|_c \leq 2.$$

The maximal length of a short string is 6. (For example,  $a^4 c^2$  and  $b^4 c^2$  are short strings of length 6.) We also call a pair of strings  $(v_1, v_2)$  *long* (or *short*) if  $v_1 v_2$  is long (or short, respectively).

According to the definition of an  $n$ -decomposition, a leaf node in a 2-decomposition

<sup>7</sup>This fact was first verified by the computer program accompanying this paper. The program, written in C, implements a generic, memoized top-down recognizer for the language  $\{w \in \text{MIX} \mid w \text{ has a 2-decomposition}\}$ , and does not rely on any special properties of the string  $z$ .

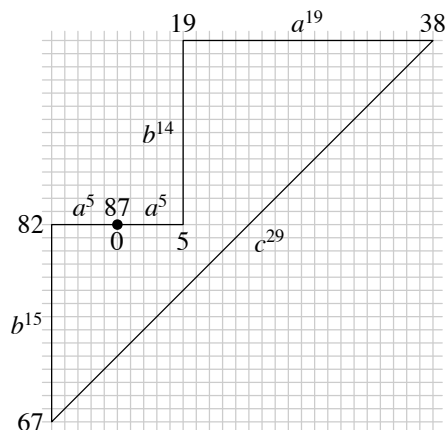


Figure 2: Graphical representation of the string  $z = a^5 b^{14} a^{19} c^{29} b^{15} a^5$ . Note that every point  $(i, j)$  on the diagonal segment has  $i > 7$  or  $j < -2$ .

must be labeled by a short pair of strings. We call a 2-decomposition *normal* if the label of every internal node is long. Clearly, any 2-decomposition can be turned into a normal 2-decomposition by deleting all nodes that are descendants of nodes with short labels.

One important property of the string  $z$  is the following:

**Lemma 5.** *If  $z = x_1 v x_2$  and  $\psi(v) \in [-2, 2] \times [-2, 2]$ , then either  $v$  or  $x_1 x_2$  is short.*

*Proof.* This is easy to see from the graphical representation in Figure 2. If a substring  $v$  of  $z$  has  $\psi(v) \in [-2, 2] \times [-2, 2]$ , then the subcurve corresponding to  $v$  must have initial and final coordinates whose difference lies in  $[-2, 2] \times [-2, 2]$ . If  $v$  contains all three letters, then it must contain as a substring at least one of  $ba^{19}c$ ,  $ac^{29}b$ , and  $cb^{15}a$ . The only way to satisfy both these conditions is to have the subcurve corresponding to  $v$  start and end very close to the origin, so that  $x_1 x_2$  is short. (Note that the distance between the coordinate  $(5, 0)$  corresponding to position 5 of  $z$  and the diagonal segment corresponding to the substring  $c^{29}$  is large enough that it is impossible for  $v$  to start at position 5 and end in the middle of  $c^{29}$  without violating the condition  $\psi(v) \in [-2, 2] \times [-2, 2]$ .) □

Lemma 5 leads to the following observation. Let us call a decomposition of a string *concatenation-free* if each of its non-leaf labels is the wrapping of the labels of the children.

**Lemma 6.** *If  $z$  has a 2-decomposition, then  $z$  has a normal, concatenation-free 2-decomposition.*

*Proof.* Let  $\mathcal{D}$  be a 2-decomposition of  $z$ . Without loss of generality, we may assume that  $\mathcal{D}$  is normal. Suppose that  $\mathcal{D}$  contains a node  $\mu$  whose label is the left or right concatenation of the labels of its children,  $(u_1, u_2)$  and  $(v_1, v_2)$ . We only consider the case of left concatenation since the case of right concatenation is entirely analogous; so we suppose that the node  $\mu$  is labeled by  $(u_1u_2v_1, v_2)$ . It follows that  $z = x_1u_1u_2x_2$  for some  $x_1, x_2$ , and by Lemma 5, either  $u_1u_2$  or  $x_1x_2$  is short. If  $u_1u_2$  is short, then the left child of  $\mu$  is a leaf because  $\mathcal{D}$  is normal. We can replace its label by  $(u_1u_2, \varepsilon)$ ; the label  $(u_1u_2v_1, v_2)$  of  $\mu$  will now be the wrapping (as well as left concatenation) of the two child labels,  $(u_1u_2, \varepsilon)$  and  $(v_1, v_2)$ . If  $x_1x_2$  is short, then we can combine by wrapping a single node labeled by  $(x_1, x_2)$  with the subtree of  $\mathcal{D}$  rooted at the left child of  $\mu$ , to obtain a new 2-decomposition of  $z$ . In either case, the result is a normal 2-decomposition of  $z$  with fewer instances of concatenation. Repeating this procedure, we eventually obtain a normal, concatenation-free 2-decomposition of  $z$ .  $\square$

Another useful property of the string  $z$  is the following:

**Lemma 7.** *Suppose that the following conditions hold:*

- (i)  $z = x_1u_1v_1yv_2u_2x_2$ ,
- (ii)  $x_1yx_2$  is a short string, and
- (iii) both  $\psi(u_1u_2)$  and  $\psi(v_1v_2)$  are in  $[-2, 2] \times [-2, 2]$ .

*Then either  $(u_1, u_2)$  or  $(v_1, v_2)$  is short.*

*Proof.* Suppose  $(u_1, u_2)$  and  $(v_1, v_2)$  are both long. Since  $(u_1, u_2)$  and  $(v_1, v_2)$  must both contain  $c$ , either  $u_1$  ends in  $c$  and  $v_1$  starts in  $c$ , or else  $v_2$  ends in  $c$  and  $u_2$  starts in  $c$ .

*Case 1.*  $u_1$  ends in  $c$  and  $v_1$  starts in  $c$ . Since  $(v_1, v_2)$  must contain at least one occurrence of  $a$ , the string  $v_1yv_2$  must contain  $cb^{15}a$  as a substring.

$$\begin{array}{|c|c|c|c|c|} \hline a^5b^{14} & a^{19} & c^{29} & b^{15} & a^5 \\ \hline \end{array}$$

$\underbrace{\hspace{10em}}_{v_1yv_2}$

Since  $x_1yx_2$  is short, we have  $|y|_b \leq 4$ . It follows that  $|v_1v_2|_b \geq 11$ . But  $v_1yv_2$  is a substring of  $c^{28}b^{15}a^5$ , so  $|v_1v_2|_a \leq 5$ . This clearly contradicts  $\psi(v_1v_2) \in [-2, 2] \times [-2, 2]$ .

*Case 2.*  $v_2$  ends in  $c$  and  $u_2$  starts in  $c$ . In this case,  $cb^{15}a^5$  is a suffix of  $u_2x_2$ . Since  $x_1yx_2$  is short,  $|x_2|_a \leq 4$ . This means that  $cb^{15}a$  is a substring of  $u_2$  and hence  $|u_2|_b = 15$ .

$$\begin{array}{|c|c|c|c|c|} \hline a^5b^{14} & a^{19} & c^{29} & b^{15} & a^5 \\ \hline \end{array}$$

$\underbrace{\hspace{10em}}_{v_1yv_2}$

On the other hand, since  $(v_1, v_2)$  must contain at least one occurrence of  $b$ , the string  $v_1yv_2$  must contain  $ba^{19}c$  as a substring. This implies that  $|u_1u_2|_a \leq 10$ . But since  $|u_2|_b = 15$ , we have  $|u_1u_2|_b \geq 15$ . This clearly contradicts  $\psi(u_1u_2) \in [-2, 2] \times [-2, 2]$ .  $\square$

We now assume that  $z$  has a normal, concatenation-free 2-decomposition  $\mathcal{D}$  and derive a contradiction. We do this by following a certain path in  $\mathcal{D}$ . Starting from the root, we descend in  $\mathcal{D}$ , always choosing a non-leaf child, as long as there is one. We show that this path will never terminate.

The  $i$ -th node on the path will be denoted by  $\mu_i$ , counting the root as the 0-th node. The label of  $\mu_i$  will be denoted by  $(w_{i,1}, w_{i,2})$ . With each  $i$ , we associate three strings  $x_{i,1}, y_i, x_{i,2}$  such that  $x_{i,1}w_{i,1}y_iw_{i,2}x_{i,2} = z$ , analogously to Lemma 1. Since  $\psi(w_{i,1}w_{i,2}) \in [-2, 2] \times [-2, 2]$  and  $\psi(z) = (0, 0)$ , we will always have  $\psi(x_{i,1}y_ix_{i,2}) \in [-2, 2] \times [-2, 2]$ .

Initially,  $(w_{0,1}, w_{0,2})$  is the label of the root  $\mu_0$  and  $x_{0,1} = y_0 = x_{0,2} = \varepsilon$ . If  $\mu_i$  is not a leaf node, let  $(u_{i,1}, u_{i,2})$  and  $(v_{i,1}, v_{i,2})$  be the labels of the left and right children of  $\mu_i$ , respectively. If the left child is not a leaf node, we let  $\mu_{i+1}$  be the left child, in which case we have  $(w_{i+1,1}, w_{i+1,2}) = (u_{i,1}, u_{i,2})$ ,  $x_{i+1,1} = x_{i,1}$ ,  $x_{i+1,2} = x_{i,2}$ , and  $y_{i+1} = v_{i,1}yv_{i,2}$ . Otherwise,  $\mu_{i+1}$  will be the right child of  $\mu_i$ , and we have  $(w_{i+1,1}, w_{i+1,2}) = (v_{i,1}, v_{i,2})$ ,  $x_{i+1,1} = x_{i,1}u_{i,1}$ ,  $x_{i+1,2} = u_{i,2}x_{i,2}$ , and  $y_{i+1} = y_i$ .

The path  $\mu_0, \mu_1, \mu_2, \dots$  is naturally divided into two parts. The initial part of the path consists of nodes where  $x_{i,1}y_ix_{i,2}$  is short. Note that  $x_{0,1}y_0x_{0,2} = \varepsilon$  is short. As long as  $x_{i,1}y_ix_{i,2}$  is short,  $(w_{i,1}, w_{i,2})$  must be long and  $\mu_i$  has two children labeled by  $(u_{i,1}, u_{i,2})$  and  $(v_{i,1}, v_{i,2})$ . By Lemma 7, either  $(u_{i,1}, u_{i,2})$  or  $(v_{i,1}, v_{i,2})$  must be short. Since the length

of  $z$  is 87 and the length of a short string is at most 6, exactly one of  $(u_{i,1}, u_{i,2})$  and  $(v_{i,1}, v_{i,2})$  must be long.

We must eventually enter the second part of the path, where  $x_{i,1}y_ix_{i,2}$  is no longer short. Let  $\mu_m$  be the first node belonging to this part of the path. Note that at  $\mu_m$ , we have  $\psi(x_{m,1}y_mx_{m,2}) = \psi(x_{m-1,1}y_{m-1}x_{m-1,2}) + \psi(v)$  for some short string  $v$ . (Namely,  $v = u_{m-1,1}u_{m-1,2}$  or  $v = v_{m-1,1}v_{m-1,2}$ .)

**Lemma 8.** *If  $u$  and  $v$  are short strings and  $\psi(uv) \in [-2, 2] \times [-2, 2]$ , then  $|uv|_d \leq 4$  for each  $d \in \{a, b, c\}$ .*

*Proof.* Since  $u$  and  $v$  are short, we have  $|u|_a \leq 4, |u|_b \leq 4, |u|_c \leq 2$  and  $|v|_a \leq 4, |v|_b \leq 4, |v|_c \leq 2$ . It immediately follows that  $|uv|_c \leq 4$ . We distinguish two cases.

*Case 1.*  $|uv|_c \leq 2$ . Since  $\psi(uv) \in [-2, 2] \times [-2, 2]$ , we must have  $|uv|_a \leq 4$  and  $|uv|_b \leq 4$ .

*Case 2.*  $|uv|_c \geq 3$ . Since  $|u|_c \leq 2$  and  $|v|_c \leq 2$ , we must have  $|u|_c \geq 1$  and  $|v|_c \geq 1$ . Also,  $\psi(uv) \in [-2, 2] \times [-2, 2]$  implies that  $|uv|_a \geq 1$  and  $|uv|_b \geq 1$ . Since  $u$  and  $v$  are short, it follows that one of the following two conditions must hold:

- (i)  $|u|_a \geq 1, |u|_b = 0$  and  $|v|_a = 0, |v|_b \geq 1$ .
- (ii)  $|u|_a = 0, |u|_b \geq 1$  and  $|v|_a \geq 1, |v|_b = 0$ .

In the former case,  $|uv|_a = |u|_a \leq 4$  and  $|uv|_b = |v|_b \leq 4$ . In the latter case,  $|uv|_a = |v|_a \leq 4$  and  $|uv|_b = |u|_b \leq 4$ .  $\square$

By Lemma 8, the number of occurrences of each letter in  $x_{m,1}y_mx_{m,2}$  is in  $[1, 4]$ . This can only be if

$$\begin{aligned} x_{m,1}x_{m,2} &= a^j, \\ y_m &= c^k b^l, \end{aligned}$$

for some  $j, k, l \in [1, 4]$ . This means that the string  $z$  must have been split into two strings  $(w_{0,1}, w_{0,2})$  at the root of  $\mathcal{D}$  somewhere in the vicinity of position 67 (see Figure 2).

It immediately follows that for all  $i \geq m$ ,  $w_{i,1}$  is a substring of  $a^5 b^{14} a^{19} c^{28}$  and  $w_{i,2}$  is a substring of  $b^{14} a^5$ . We show by induction that for all  $i \geq m$ , the following condition holds:

( $\dagger$ )  $ba^{19}c^{17}$  is a substring of  $w_{i,1}$ .

The condition ( $\dagger$ ) clearly holds for  $i = m$ . Now assume ( $\dagger$ ). Then  $(w_{i,1}, w_{i,2})$  is long, and  $\mu_i$  has left and

right children, labeled by  $(u_{i,1}, u_{i,2})$  and  $(v_{i,1}, v_{i,2})$ , respectively, such that  $w_{i,1} = u_{i,1}v_{i,1}$  and  $w_{i,2} = v_{i,2}u_{i,2}$ . We consider two cases.

*Case 1.*  $u_{i,1}$  contains  $c$ . Then  $ba^{19}c$  is a substring of  $u_{i,1}$ . Since  $u_{i,2}$  is a substring of  $b^{14}a^5$ , it cannot contain any occurrences of  $c$ . Since  $\psi_1(u_{i,1}u_{i,2}) \in [-2, 2]$ , it follows that  $u_{i,1}$  must contain at least 17 occurrences of  $c$ ; hence  $ba^{19}c^{17}$  is a substring of  $u_{i,1}$ . Since  $(u_{i,1}, u_{i,2})$  is long,  $(w_{i+1,1}, w_{i+1,2}) = (u_{i,1}, u_{i,2})$ . Therefore, the condition ( $\dagger$ ) holds with  $i + 1$  in place of  $i$ .

*Case 2.*  $u_{i,1}$  does not contain  $c$ . Then  $(u_{i,1}, u_{i,2})$  is short and  $(w_{i+1,1}, w_{i+1,2}) = (v_{i,1}, v_{i,2})$ . Note that  $v_{i,1}$  must contain at least 17 occurrences of  $c$ , but  $v_{i,2}$  is a substring of  $b^{14}a^5$  and hence cannot contain more than 14 occurrences of  $b$ . Since  $\psi_2(v_{i,1}v_{i,2}) \in [-2, 2]$ , it follows that  $v_{i,1}$  must contain at least one occurrence of  $b$ . Therefore,  $ba^{19}c^{17}$  must be a substring of  $v_{i,1} = w_{i+1,1}$ , which shows that ( $\dagger$ ) holds with  $i + 1$  in place of  $i$ .

We have proved that ( $\dagger$ ) holds for all  $i \geq m$ . It follows that for all  $i$ ,  $\mu_i$  has two children and hence  $\mu_{i+1}$  is defined. This means that the path  $\mu_0, \mu_1, \mu_2, \dots$  is infinite, contradicting the assumption that  $\mathcal{D}$  is a 2-decomposition of  $z$ .

We have proved the following:

**Lemma 9.** *There is a string in MIX that has no 2-decomposition.*

**Theorem 10.** *There is no head grammar  $G$  such that  $L(G) = \text{MIX}$ .*

*Proof.* Immediate from Lemmas 3, 4, and 9.  $\square$

## References

- Setsuo Arikawa, Takeshi Shinohara, and Akihiro Yamamoto. 1992. Learning elementary formal systems. *Theoretical Computer Science*, 95(1):97–113.
- Emmon Bach. 1981. Discontinuous constituents in generalized categorial grammars. In Victoria Burke and James Pustejovsky, editors, *Proceedings of the 11th Annual Meeting of the North East Linguistic Society*, pages 1–12.
- Emmon Bach. 1988. Categorial grammars as theories of language. In Richard T. Oehrle, Emmon Bach, and Deirdre Wheeler, editors, *Categorial Grammars and Natural Language Structures*, pages 17–34. D. Reidel, Dordrecht.

- Gerald Gazdar. 1988. Applicability of indexed grammars to natural languages. In U. Reyle and C. Rohrer, editors, *Natural Language Parsing and Linguistic Theories*, pages 69–94. D. Reidel Publishing Company, Dordrecht.
- Robert Gilman. 2005. Formal languages and their application to combinatorial group theory. In Alexandre V. Borovik, editor, *Groups, Languages, Algorithms*, number 378 in Contemporary Mathematics, pages 1–36. American Mathematical Society, Providence, RI.
- Annius V. Groenink. 1997. Mild context-sensitivity and tuple-based generalizations of context-free grammar. *Linguistics and Philosophy*, 20:607–636.
- Aravind K. Joshi, Vijay K. Shanker, and David J. Weir. 1991. The convergence of mildly context-sensitive grammar formalisms. In Peter Sells, Stuart M. Shieber, and Thomas Wasow, editors, *Foundational Issues in Natural Language Processing*, pages 31–81. The MIT Press, Cambridge, MA.
- Aravind K. Joshi. 1985. Tree-adjointing grammars: How much context sensitivity is required to provide reasonable structural descriptions? In David Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, Cambridge.
- Markus Kracht. 2003. *The Mathematics of Language*, volume 63 of *Studies in Generative Grammar*. Mouton de Gruyter, Berlin.
- William Marsh. 1985. Some conjectures on indexed languages. Paper presented to the Association for Symbolic Logic Meeting, Stanford University, July 15–19. Abstract appears in *Journal of Symbolic Logic* 51(3):849 (1986).
- Carl J. Pollard. 1984. *Generalized Phrase Structure Grammars, Head Grammars, and Natural Language*. Ph.D. thesis, Department of Linguistics, Stanford University.
- Kelly Roach. 1987. Formal properties of head grammars. In Alexis Manaster-Ramer, editor, *Mathematics of Language*, pages 293–347. John Benjamins, Amsterdam.
- Sylvain Salvati. 2011. MIX is a 2-MCFL and the word problem in  $\mathbb{Z}^2$  is captured by the IO and the OI hierarchies. Technical report, INRIA.
- Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context free grammars. *Theoretical Computer Science*, 88(2):191–229.
- Raymond M. Smullyan. 1961. *Theory of Formal Systems*. Princeton University Press, Princeton, NJ.
- K. Vijay-Shanker and D. J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory*, 27:511–546.
- K. Vijay-Shanker, David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *25th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.
- David J. Weir. 1988. *Characterizing Mildly Context-Sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

# Exploiting Multiple Treebanks for Parsing with Quasi-synchronous Grammars

Zhenghua Li, Ting Liu\*, Wanxiang Che

Research Center for Social Computing and Information Retrieval  
School of Computer Science and Technology  
Harbin Institute of Technology, China  
{lzh, tliu, car}@ir.hit.edu.cn

## Abstract

We present a simple and effective framework for exploiting multiple monolingual treebanks with different annotation guidelines for parsing. Several types of *transformation patterns* (TP) are designed to capture the systematic annotation inconsistencies among different treebanks. Based on such TPs, we design *quasi-synchronous grammar* features to augment the baseline parsing models. Our approach can significantly advance the state-of-the-art parsing accuracy on two widely used target treebanks (Penn Chinese Treebank 5.1 and 6.0) using the Chinese Dependency Treebank as the source treebank. The improvements are respectively 1.37% and 1.10% with automatic part-of-speech tags. Moreover, an indirect comparison indicates that our approach also outperforms previous work based on treebank conversion.

## 1 Introduction

The scale of available labeled data significantly affects the performance of statistical data-driven models. As a structural classification problem that is more challenging than binary classification and sequence labeling problems, syntactic parsing is more prone to suffer from the data sparseness problem. However, the heavy cost of treebanking typically limits one single treebank in both scale and genre. At present, learning from one single treebank seems inadequate for further boosting parsing accuracy.<sup>1</sup>

\*Correspondence author: tliu@ir.hit.edu.cn

<sup>1</sup>Incorporating an increased number of global features, such as third-order features in graph-based parsers, slightly affects parsing accuracy (Koo and Collins, 2010; Li et al., 2011).

Treebanks	# of Words	Grammar
CTB5	0.51 million	Phrase structure
CTB6	0.78 million	Phrase structure
CDT	1.11 million	Dependency structure
Sinica	0.36 million	Phrase structure
TCT	about 1 million	Phrase structure

Table 1: Several publicly available Chinese treebanks.

Therefore, studies have recently resorted to other resources for the enhancement of parsing models, such as large-scale unlabeled data (Koo et al., 2008; Chen et al., 2009; Bansal and Klein, 2011; Zhou et al., 2011), and bilingual texts or cross-lingual treebanks (Burkett and Klein, 2008; Huang et al., 2009; Burkett et al., 2010; Chen et al., 2010).

The existence of multiple monolingual treebanks opens another door for this issue. For example, table 1 lists a few publicly available Chinese treebanks that are motivated by different linguistic theories or applications. In the current paper, we utilize the first three treebanks, i.e., the Chinese Penn Treebank 5.1 (CTB5) and 6.0 (CTB6) (Xue et al., 2005), and the Chinese Dependency Treebank (CDT) (Liu et al., 2006). The Sinica treebank (Chen et al., 2003) and the Tsinghua Chinese Treebank (TCT) (Qiang, 2004) can be similarly exploited with our proposed approach, which we leave as future work.

Despite the divergence of annotation philosophy, these treebanks contain rich human knowledge on the Chinese syntax, thereby having a great deal of common ground. Therefore, exploiting multiple treebanks is very attractive for boosting parsing accuracy. Figure 1 gives an example with different an-

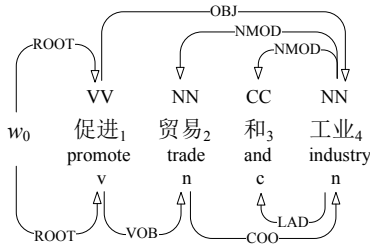


Figure 1: Example with annotations from CTB5 (upper) and CDT (under).

notations from CTB5 and CDT.<sup>2</sup> This example illustrates that the two treebanks annotate coordination constructions differently. In CTB5, the last noun is the head, whereas the first noun is the head in CDT.

One natural idea for multiple treebank exploitation is *treebank conversion*. First, the annotations in the source treebank are converted into the style of the target treebank. Then, both the converted treebank and the target treebank are combined. Finally, the combined treebank are used to train a better parser. However, the inconsistencies among different treebanks are normally nontrivial, which makes rule-based conversion infeasible. For example, a number of inconsistencies between CTB5 and CDT are lexicon-sensitive, that is, they adopt different annotations for some particular lexicons (or word senses). Niu et al. (2009) use sophisticated strategies to reduce the noises of the converted treebank after automatic treebank conversion.

The present paper proposes a simple and effective framework for this problem. The proposed framework avoids directly addressing the difficult annotation transformation problem, but focuses on modeling the annotation inconsistencies using *transformation patterns* (TP). The TPs are used to compose *quasi-synchronous grammar* (QG) features, such that the knowledge of the source treebank can inspire the target parser to build better trees. We conduct extensive experiments using CDT as the source treebank to enhance two target treebanks (CTB5 and CTB6). Results show that our approach can significantly boost state-of-the-art parsing accuracy. Moreover, an indirect comparison indicates that our ap-

<sup>2</sup>CTB5 is converted to dependency structures following the standard practice of dependency parsing (Zhang and Clark, 2008b). Notably, converting a phrase-structure tree into its dependency-structure counterpart is straightforward and can be performed by applying heuristic head-finding rules.

proach also outperforms the treebank conversion approach of Niu et al. (2009).

## 2 Related Work

The present work is primarily inspired by Jiang et al. (2009) and Smith and Eisner (2009). Jiang et al. (2009) improve the performance of word segmentation and part-of-speech (POS) tagging on CTB5 using another large-scale corpus of different annotation standards (People’s Daily). Their framework is similar to ours. However, handling syntactic annotation inconsistencies is significantly more challenging in our case of parsing. Smith and Eisner (2009) propose effective QG features for parser adaptation and projection. The first part of their work is closely connected with our work, but with a few important differences. First, they conduct simulated experiments on one treebank by manually creating a few trivial annotation inconsistencies based on two heuristic rules. They then focus on better adapting a parser to a new annotation style with few sentences of the target style. In contrast, we experiment with two real large-scale treebanks, and boost the state-of-the-art parsing accuracy using QG features. Second, we explore much richer QG features to fully exploit the knowledge of the source treebank. These features are tailored to the dependency parsing problem. In summary, the present work makes substantial progress in modeling structural annotation inconsistencies with QG features for parsing.

Previous work on treebank conversion primarily focuses on converting one grammar formalism of a treebank into another and then conducting a study on the converted treebank (Collins et al., 1999; Xia et al., 2008). The work by Niu et al. (2009) is, to our knowledge, the only study to date that combines the converted treebank with the existing target treebank. They automatically convert the dependency-structure CDT into the phrase-structure style of CTB5 using a statistical constituency parser trained on CTB5. Their experiments show that the combined treebank can significantly improve the performance of constituency parsers. However, their method requires several sophisticated strategies, such as corpus weighting and score interpolation, to reduce the influence of conversion errors. Instead of using the noisy converted treebank as additional training data, our approach allows the QG-



enhanced parsing models to softly learn the systematic inconsistencies based on QG features, making our approach simpler and more robust.

Our approach is also intuitively related to *stacked learning* (SL), a machine learning framework that has recently been applied to dependency parsing to integrate two main-stream parsing models, i.e., graph-based and transition-based models (Nivre and McDonald, 2008; Martins et al., 2008). However, the SL framework trains two parsers on the same treebank and therefore does not need to consider the problem of annotation inconsistencies.

### 3 Dependency Parsing

Given an input sentence  $\mathbf{x} = w_0 w_1 \dots w_n$  and its POS tag sequence  $\mathbf{t} = t_0 t_1 \dots t_n$ , the goal of dependency parsing is to build a dependency tree as depicted in Figure 1, denoted by  $\mathbf{d} = \{(h, m, l) : 0 \leq h \leq n, 0 < m \leq n, l \in \mathcal{L}\}$ , where  $(h, m, l)$  indicates an directed arc from the *head word* (also called *father*)  $w_h$  to the *modifier* (also called *child* or *dependent*)  $w_m$  with a dependency label  $l$ , and  $\mathcal{L}$  is the label set. We omit the label  $l$  because we focus on unlabeled dependency parsing in the present paper. The artificial node  $w_0$ , which always points to the root of the sentence, is used to simplify the formalizations.

In the current research, we adopt the graph-based parsing models for their state-of-the-art performance in a variety of languages.<sup>3</sup> Graph-based models view the problem as finding the highest scoring tree from a directed graph. To guarantee the efficiency of the decoding algorithms, the score of a dependency tree is factored into the scores of some small parts (subtrees).

$$\begin{aligned} \text{Score}_{bs}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \mathbf{w}_{bs} \cdot \mathbf{f}_{bs}(\mathbf{x}, \mathbf{t}, \mathbf{d}) \\ &= \sum_{p \subseteq \mathbf{d}} \mathbf{w}_{part} \cdot \mathbf{f}_{part}(\mathbf{x}, \mathbf{t}, p) \end{aligned}$$

where  $p$  is a scoring part which contains one or more dependencies of  $\mathbf{d}$ , and  $\mathbf{f}_{bs}(\cdot)$  denotes the *basic parsing features*, as opposed to the *QG features*. Figure 2 lists the scoring parts used in our work, where  $g$ ,  $h$ ,  $m$ , and  $s$ , are word indices.

We implement three parsing models of varying strengths in capturing features to better understand the effect of the proposed QG features.

<sup>3</sup>Our approach can equally be applied to transition-based parsing models (Yamada and Matsumoto, 2003; Nivre, 2003) with minor modifications.

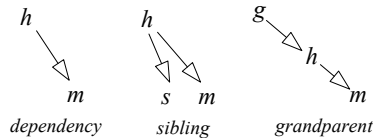


Figure 2: Scoring parts used in our graph-based parsing models.

- **The first-order model (O1)** only incorporates dependency parts (McDonald et al., 2005), and requires  $O(n^3)$  parsing time.
- **The second-order model using only sibling parts (O2sib)** includes both dependency and sibling parts (McDonald and Pereira, 2006), and needs  $O(n^3)$  parsing time.
- **The second-order model (O2)** uses all the scoring parts in Figure 2 (Koo and Collins, 2010). The time complexity of the decoding algorithm is  $O(n^4)$ .

For the O2 model, the score function is rewritten as:

$$\begin{aligned} \text{Score}_{bs}(\mathbf{x}, \mathbf{t}, \mathbf{d}) &= \sum_{\{(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{dep} \cdot \mathbf{f}_{dep}(\mathbf{x}, \mathbf{t}, h, m) \\ &+ \sum_{\{(h,s),(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{sib} \cdot \mathbf{f}_{sib}(\mathbf{x}, \mathbf{t}, h, s, m) \\ &+ \sum_{\{(g,h),(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{grad} \cdot \mathbf{f}_{grad}(\mathbf{x}, \mathbf{t}, g, h, m) \end{aligned}$$

where  $\mathbf{f}_{dep}(\cdot)$ ,  $\mathbf{f}_{sib}(\cdot)$  and  $\mathbf{f}_{grad}(\cdot)$  correspond to the features for the three kinds of scoring parts. We adopt the standard features following Li et al. (2011). For the O1 and O2sib models, the above formula is modified by deactivating the extra parts.

### 4 Dependency Parsing with QG Features

Smith and Eisner (2006) propose the QG for machine translation (MT) problems, allowing greater syntactic divergences between the two languages. Given a source sentence  $\mathbf{x}'$  and its syntactic tree  $\mathbf{d}'$ , a QG defines a monolingual grammar that generates translations of  $\mathbf{x}'$ , which can be denoted by  $p(\mathbf{x}, \mathbf{d}, \mathbf{a} | \mathbf{x}', \mathbf{d}')$ , where  $\mathbf{x}$  and  $\mathbf{d}$  refer to a translation and its parse, and  $\mathbf{a}$  is a cross-language alignment. Under a QG, any portion of  $\mathbf{d}$  can be aligned to any

<sup>4</sup>We use the coarse-to-fine strategy to prune the search space, which largely accelerates the decoding procedure (Koo and Collins, 2010).

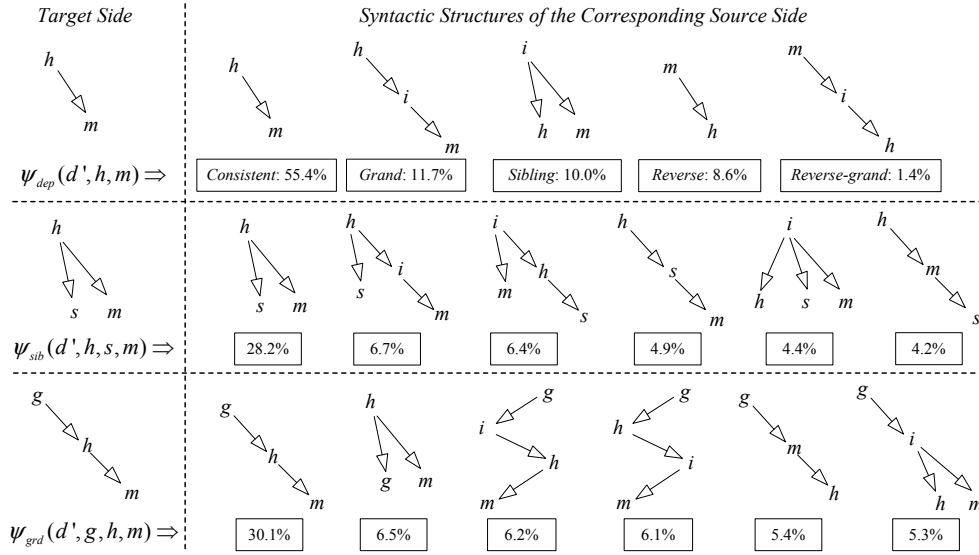


Figure 4: Most frequent transformation patterns (TPs) when using CDT as the source treebank and CTB5 as the target. A TP comprises two syntactic structures, one in the source side and the other in the target side, and denotes the process by which the left-side subtree is transformed into the right-side structure. Functions  $\psi_{dep}(\cdot)$ ,  $\psi_{sib}(\cdot)$ , and  $\psi_{grd}(\cdot)$  return the specific TP type for a candidate scoring part according to the source tree  $d'$ .

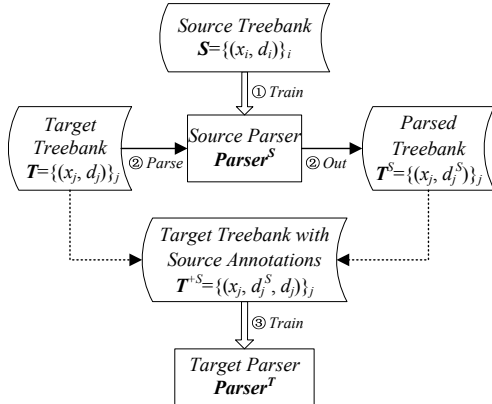


Figure 3: Framework of our approach.

portion of  $d'$ , and the construction of  $d$  can be inspired by arbitrary substructures of  $d'$ . To date, QGs have been successfully applied to various tasks, such as word alignment (Smith and Eisner, 2006), machine translation (Gimpel and Smith, 2011), question answering (Wang et al., 2007), and sentence simplification (Woodsend and Lapata, 2011).

In the present work, we utilize the idea of the QG for the exploitation of multiple monolingual treebanks. The key idea is to let the parse tree of one style inspire the parsing process of another style. Different from a MT process, our problem consid-

ers one single sentence ( $\mathbf{x} = \mathbf{x}'$ ), and the alignment  $\mathbf{a}$  is trivial. Figure 3 shows the framework of our approach. First, we train a statistical parser on the source treebank, which is called the source parser. The source parser is then used to parse the whole target treebank. At this point, the target treebank contains two sets of annotations, one conforming to the source style, and the other conforming to the target style. During both the training and test phases, the target parser are inspired by the source annotations, and the score of a target dependency tree becomes

$$Score(\mathbf{x}, \mathbf{t}, \mathbf{d}', \mathbf{d}) = Score_{bs}(\mathbf{x}, \mathbf{t}, \mathbf{d}) + Score_{qg}(\mathbf{x}, \mathbf{t}, \mathbf{d}', \mathbf{d})$$

The first part corresponds to the baseline model, whereas the second part is affected by the source tree  $d'$  and can be rewritten as

$$Score_{qg}(\mathbf{x}, \mathbf{t}, \mathbf{d}', \mathbf{d}) = \mathbf{w}_{qg} \cdot \mathbf{f}_{qg}(\mathbf{x}, \mathbf{t}, \mathbf{d}', \mathbf{d})$$

where  $\mathbf{f}_{qg}(\cdot)$  denotes the QG features. We expect the QG features to encourage or penalize certain scoring parts in the target side according to the source tree  $d'$ . Taking Figure 1 as an example, suppose that the upper structure is the target. The target parser can raise the score of the candidate dependence “and”  $\leftarrow$  “industry”, because the depen-

dependency also appears in the source structure, and evidence in the training data shows that both annotation styles handle conjunctions in the same manner. Similarly, the parser may add weight to “*trade*” ← “*industry*”, considering that the *reverse* arc is in the source structure. Therefore, the QG-enhanced model must learn the systematic consistencies and inconsistencies from the training data.

To model such consistency or inconsistency systematicness, we propose the use of TPs for encoding the structural correspondence between the source and target styles. Figure 4 presents the three kinds of TPs used in our model, which correspond to the three scoring parts of our parsing models.

Dependency TPs shown in the first row consider how one dependency in the target side is transformed in the source annotations. We only consider the five cases shown in the figure. The percentages in the lower boxes refer to the proportion of the corresponding pattern, which are counted from the training data of the target treebank with source annotations  $T^{+S}$ . We can see that the noisy source structures and the gold-standard target structures have 55.4% common dependencies. If the source structure does not belong to any of the listed five cases,  $\psi_{dep}(\mathbf{d}', h, m)$  returns “*else*” (12.9%). We could consider more complex structures, such as  $h$  being the grand grand father of  $m$ , but statistics show that more complex transformations become very scarce in the training data.

For the reason that dependency TPs can only model how one dependency in the target structure is transformed, we consider more complex transformations for the other two kinds of scoring parts of the target parser, i.e., the sibling and grand TPs shown in the bottom two rows. We only use high-frequency TPs of a proportion larger than 1.0%, aggregate others as “*else*”, which leaves us with 21 sibling TPs and 22 grand TPs.

Based on these TPs, we propose the QG features for enhancing the baseline parsing models, which are shown in Table 2. The type of the TP is conjoined with the related words and POS tags, such that the QG-enhanced parsing models can make more elaborate decisions based on the context. Then, the score contributed by the QG features can

be redefined as

$$\begin{aligned} Score_{qg}(\mathbf{x}, \mathbf{t}, \mathbf{d}', \mathbf{d}) = & \sum_{\{(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{qg-dep} \cdot \mathbf{f}_{qg-dep}(\mathbf{x}, \mathbf{t}, \mathbf{d}', h, m) \\ & + \sum_{\{(h,s),(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{qg-sib} \cdot \mathbf{f}_{qg-sib}(\mathbf{x}, \mathbf{t}, \mathbf{d}', h, s, m) \\ & + \sum_{\{(g,h),(h,m)\} \subseteq \mathbf{d}} \mathbf{w}_{qg-grd} \cdot \mathbf{f}_{qg-grd}(\mathbf{x}, \mathbf{t}, \mathbf{d}', g, h, m) \end{aligned}$$

which resembles the baseline model and can be naturally handled by the decoding algorithms.

## 5 Experiments and Analysis

We use the CDT as the source treebank (Liu et al., 2006). CDT consists of 60,000 sentences from the People’s Daily in 1990s. For the target treebank, we use two widely used versions of Penn Chinese Treebank, i.e., CTB5 and CTB6, which consist of Xinhua newswire, Hong Kong news and articles from Sinarama news magazine (Xue et al., 2005). To facilitate comparison with previous results, we follow Zhang and Clark (2008b) for data split and constituency-to-dependency conversion of CTB5. CTB6 is used as the Chinese data set in the CoNLL 2009 shared task (Hajič et al., 2009). Therefore, we adopt the same setting.

CDT and CTB5/6 adopt different POS tag sets, and converting from one tag set to another is difficult (Niu et al., 2009).<sup>5</sup> To overcome this problem, we use the People’s Daily corpus (PD),<sup>6</sup> a large-scale corpus annotated with word segmentation and POS tags, to train a statistical POS tagger. The tagger produces a universal layer of POS tags for both the source and target treebanks. Based on the common tags, the source parser projects the source annotations into the target treebanks. PD comprises approximately 300 thousand sentences of with approximately 7 million words from the first half of 1998 of People’s Daily.

Table 3 summarizes the data sets used in the present work. CTB5X is the same with CTB5 but follows the data split of Niu et al. (2009). We use CTB5X to compare our approach with their treebank conversion method (see Table 9).

<sup>5</sup>The word segmentation standards of the two treebanks also slightly differs, which are not considered in this work.

<sup>6</sup>[http://iccl.pku.edu.cn/iccl\\_groups/corpus tagging.asp](http://iccl.pku.edu.cn/iccl_groups/corpus tagging.asp)

$\mathbf{f}_{qg-dep}(\mathbf{x}, \mathbf{t}, \mathbf{d}', h, m)$ $\oplus \text{dir}(h, m) \circ \text{dist}(h, m)$	$\mathbf{f}_{qg-sib}(\mathbf{x}, \mathbf{t}, \mathbf{d}', h, s, m)$ $\oplus \text{dir}(h, m)$	$\mathbf{f}_{qg-grd}(\mathbf{x}, \mathbf{t}, \mathbf{d}', g, h, m)$ $\oplus \text{dir}(h, m) \circ \text{dir}(g, h)$
$\psi_{dep}(\mathbf{d}', h, m) \circ t_h \circ t_m$	$\psi_{sib}(\mathbf{d}', h, s, m) \circ t_h \circ t_s \circ t_m$	$\psi_{grd}(\mathbf{d}', g, h, m) \circ t_g \circ t_h \circ t_m$
$\psi_{dep}(\mathbf{d}', h, m) \circ w_h \circ t_m$	$\psi_{sib}(\mathbf{d}', h, s, m) \circ w_h \circ t_s \circ t_m$	$\psi_{grd}(\mathbf{d}', g, h, m) \circ w_g \circ t_h \circ t_m$
$\psi_{dep}(\mathbf{d}', h, m) \circ t_h \circ w_m$	$\psi_{sib}(\mathbf{d}', h, s, m) \circ t_h \circ w_s \circ t_m$	$\psi_{grd}(\mathbf{d}', g, h, m) \circ t_g \circ w_h \circ t_m$
$\psi_{dep}(\mathbf{d}', h, m) \circ w_h \circ w_m$	$\psi_{sib}(\mathbf{d}', h, s, m) \circ t_h \circ t_s \circ w_m$	$\psi_{grd}(\mathbf{d}', g, h, m) \circ t_g \circ t_h \circ w_m$
	$\psi_{sib}(\mathbf{d}', h, s, m) \circ t_s \circ t_m$	$\psi_{grd}(\mathbf{d}', g, h, m) \circ t_g \circ t_m$

Table 2: QG features used to enhance the baseline parsing models.  $\text{dir}(h, m)$  denotes the direction of the dependency  $(h, m)$ , whereas  $\text{dist}(h, m)$  is the distance  $|h - m|$ .  $\oplus \text{dir}(h, m) \circ \text{dist}(h, m)$  indicates that the features listed in the corresponding column are also conjoined with  $\text{dir}(h, m) \circ \text{dist}(h, m)$  to form new features.

Corpus	Train	Dev	Test
PD	281,311	5,000	10,000
CDT	55,500	1,500	3,000
CTB5	16,091	803	1,910
CTB5X	18,104	352	348
CTB6	22,277	1,762	2,556

Table 3: Data used in this work (in sentence number).

We adopt *unlabeled attachment score* (UAS) as the primary evaluation metric. We also use *Root accuracy* (RA) and *complete match rate* (CM) to give more insights. All metrics exclude punctuation. We adopt Dan Bikel’s randomized parsing evaluation comparator for significance test (Noreen, 1989).<sup>7</sup>

For all models used in current work (POS tagging and parsing), we adopt averaged perceptron to train the feature weights (Collins, 2002). We train each model for 10 iterations and select the parameters that perform best on the development set.

## 5.1 Preliminaries

This subsection describes how we project the source annotations into the target treebanks. First, we train a statistical POS tagger on the training set of PD, which we name *Tagger<sup>PD</sup>*.<sup>8</sup> The tagging accuracy on the test set of PD is 98.30%.

We then use *Tagger<sup>PD</sup>* to produce POS tags for all the treebanks (CDT, CTB5, and CTB6).

Based on the common POS tags, we train a second-order source parser (O2) on CDT, denoted by *Parser<sup>CDT</sup>*. The UAS on CDT-test is 84.45%. We then use *Parser<sup>CDT</sup>* to parse CTB5 and CTB6.

<sup>7</sup>[http://www.cis.upenn.edu/\[normal-wave~\]dbikel/software.html](http://www.cis.upenn.edu/[normal-wave~]dbikel/software.html)

<sup>8</sup>We adopt the Chinese-oriented POS tagging features proposed in Zhang and Clark (2008a).

Models	without QG	with QG
O2	86.13	86.44 (+0.31, $p = 0.06$ )
O2sib	85.63	86.17 (+0.54, $p = 0.003$ )
O1	83.16	84.40 (+1.24, $p < 10^{-5}$ )
Li11	86.18	—
Z&N11	86.00	—

Table 4: Parsing accuracy (UAS) comparison on CTB5-test with gold-standard POS tags. Li11 refers to the second-order graph-based model of Li et al. (2011), whereas Z&N11 is the feature-rich transition-based model of Zhang and Nivre (2011).

At this point, both CTB5 and CTB6 contain dependency structures conforming to the style of CDT.

## 5.2 CTB5 as the Target Treebank

Table 4 shows the results when the gold-standard POS tags of CTB5 are adopted by the parsing models. We aim to analyze the efficacy of QG features under the ideal scenario wherein the parsing models suffer from no error propagation of POS tagging. We determine that our baseline O2 model achieves comparable accuracy with the state-of-the-art parsers. We also find that QG features can boost the parsing accuracy by a large margin when the baseline parser is weak (O1). The improvement shrinks for stronger baselines (O2sib and O2). This phenomenon is understandable. When gold-standard POS tags are available, the baseline features are very reliable and the QG features becomes less helpful for more complex models. The p-values in parentheses present the statistical significance of the improvements.

We then turn to the more realistic scenario wherein the gold-standard POS tags of the target treebank are unavailable. We train a POS tagger on the training set of CTB5 to produce the automatic



Models	without QG	with QG
O2	79.67	81.04 (+1.37)
O2sib	79.25	80.45 (+1.20)
O1	76.73	79.04 (+2.31)
Li11 joint	80.79	—
Li11 pipeline	79.29	—

Table 5: Parsing accuracy (UAS) comparison on CTB5-test with automatic POS tags. The improvements shown in parentheses are all statistically significant ( $p < 10^{-5}$ ).

Setting	UAS	CM	RA
$\mathbf{f}_{bs}(\cdot)$	79.67	26.81	73.82
$\mathbf{f}_{qg}(\cdot)$	79.15	26.34	74.71
$\mathbf{f}_{bs}(\cdot) + \mathbf{f}_{qg}(\cdot)$	81.04	29.63	77.17
$\mathbf{f}_{bs}(\cdot) + \mathbf{f}_{qg-dep}(\cdot)$	80.82	28.80	76.28
$\mathbf{f}_{bs}(\cdot) + \mathbf{f}_{qg-sib}(\cdot)$	80.86	28.48	76.18
$\mathbf{f}_{bs}(\cdot) + \mathbf{f}_{qg-grd}(\cdot)$	80.88	28.90	76.34

Table 6: Feature ablation for Parser-O2 on CTB5-test with automatic POS tags.

POS tags for the development and test sets of CTB5. The tagging accuracy is 93.88% on the test set. The automatic POS tags of the training set are produced using 10-fold cross-validation.<sup>9</sup>

Table 5 shows the results. We find that QG features result in a surprisingly large improvement over the O1 baseline and can also boost the state-of-the-art parsing accuracy by a large margin. Li et al. (2011) show that a joint POS tagging and dependency parsing model can significantly improve parsing accuracy over a pipeline model. Our QG-enhanced parser outperforms their best joint model by 0.25%. Moreover, the QG features can be used to enhance a joint model and achieve higher accuracy, which we leave as future work.

### 5.3 Analysis Using Parser-O2 with AUTO-POS

We then try to gain more insights into the effect of the QG features through detailed analysis. We select the state-of-the-art O2 parser and focus on the realistic scenario with automatic POS tags.

Table 6 compares the efficacy of different feature sets. The first major row analyzes the efficacy of

<sup>9</sup>We could use the POS tags produced by *Tagger<sup>PD</sup>* in Section 5.1, which however would make it difficult to compare our results with previous ones. Moreover, inferior results may be gained due to the differences between CTB5 and PD in word segmentation standards and text sources.

the basic features  $\mathbf{f}_{bs}(\cdot)$  and the QG features  $\mathbf{f}_{qg}(\cdot)$ . When using the few QG features in Table 2, the accuracy is very close to that when using the basic features. Moreover, using both features generates a large improvement. The second major row compares the efficacy of the three kinds of QG features corresponding to the three types of scoring parts. We can see that the three feature sets are similarly effective and yield comparable accuracies. Combining these features generate an additional improvement of approximately 0.2%. These results again demonstrate that all the proposed QG features are effective.

Figure 5 describes how the performance varies when the scale of CTB5 and CDT changes. In the left subfigure, the parsers are trained on part of the CTB5-train, and “16” indicates the use of all the training instances. Meanwhile, the source parser  $Parser^{CDT}$  is trained on the whole CDT-train. We can see that QG features render larger improvement when the target treebank is of smaller scale, which is quite reasonable. More importantly, the curves indicate that **a QG-enhanced parser trained on a target treebank of 16,000 sentences may achieve comparable accuracy with a baseline parser trained on a treebank that is double the size (32,000)**, which is very encouraging.

In the right subfigure, the target treebank is trained on the whole CTB5-train, whereas the source parser is trained on part of the CDT-train, and “55.5” indicates the use of all. The curve clearly demonstrates that the QG features are more helpful when the source treebank gets larger, which can be explained as follows. A larger source treebank can teach a source parser of higher accuracy; then, the better source parser can parse the target treebank more reliably; and finally, the target parser can better learn the annotation divergences based on QG features. These results demonstrate the effectiveness and stability of our approach.

Table 7 presents the detailed effect of the QG features on different dependency patterns. A pattern “VV  $\rightarrow$  NN” refers to a right-directed dependency with the head tagged as “VV” and the modifier tagged as “NN”. whereas “ $\leftarrow$ ” means left-directed. The “w/o QG” column shows the number of the corresponding dependency pattern that appears in the gold-standard trees but misses in the results of the baseline parser, whereas the signed figures in the “+QG” column are the changes made by the QG-

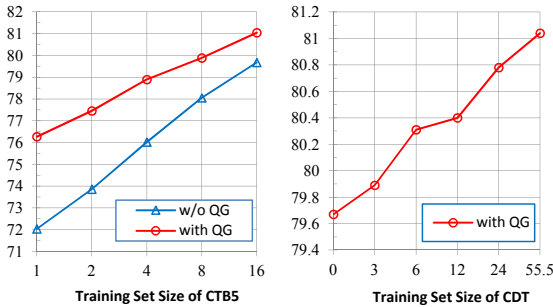


Figure 5: Parsing accuracy (UAS) comparison on CTB5-test when the scale of CDT and CTB5 varies (thousands in sentence number).

Dependency	w/o QG	+QG	Descriptions
NN ← NN	858	-78	noun modifier or coordinating nouns
VV → VV	777	-41	object clause or coordinating verbs
VV ← VV	570	-38	subject clause
VV → NN	509	-79	verb and its object
w <sub>0</sub> → VV	357	-57	verb as sentence root
VV ← NN	328	-32	attributive clause
P ← VV	278	-37	preposition phrase attachment
VV → DEC	233	-33	attributive clause and auxiliary DE
P → NN	175	-35	preposition and its object

Table 7: Detailed effect of QG features on different dependency patterns.

enhanced parser. We only list the patterns with an absolute change larger than 30. We find that the QG features can significantly help a variety of dependency patterns (i.e., reducing the missing number).

#### 5.4 CTB6 as the Target Treebank

We use CTB6 as the target treebank to further verify the efficacy of our approach. Compared with CTB5, CTB6 is of larger scale and is converted into dependency structures according to finer-grained head-finding rules (Hajič et al., 2009). We directly adopt the same transformation patterns and features tuned on CTB5. Table 8 shows results. The improvements are similar to those on CTB5, demonstrating that our approach is effective and robust. We list the top three systems of the CoNLL 2009 shared task in Table 8, showing that our approach also advances the state-of-the-art parsing accuracy on this data set.<sup>10</sup>

<sup>10</sup>We reproduce their UASs using the data released by the organizer: <http://ufal.mff.cuni.cz/conll2009-st/results/results.php>. The parsing accuracies of the top systems may be underestimated since the accuracy of the provided POS tags in CoNLL 2009 is only 92.38% on the test set, while the POS tagger used in our experiments reaches 94.08%.

Models	without QG	with QG
O2	83.23	84.33 (+1.10)
O2sib	82.87	84.11 (+1.37)
O1	80.29	82.76 (+2.47)
Bohnet (2009)	82.68	—
Che et al. (2009)	82.11	—
Gesmundo et al. (2009)	81.70	—

Table 8: Parsing accuracy (UAS) comparison on CTB6-test with automatic POS tags. The improvements shown in parentheses are all statistically significant ( $p < 10^{-5}$ ).

Models	baseline	with another treebank
Ours	84.16	86.67 (+2.51)
GP (Niu et al., 2009)	82.42	84.06 (+1.64)

Table 9: Parsing accuracy (UAS) comparison on the test set of CTB5X. Niu et al. (2009) use the maximum entropy inspired generative parser (GP) of Charniak (2000) as their constituent parser.

#### 5.5 Comparison with Treebank Conversion

As discussed in Section 2, Niu et al. (2009) automatically convert the dependency-structure CDT to the phrase-structure annotation style of CTB5X and use the converted treebank as additional labeled data. We convert their phrase-structure results on CTB5X-test into dependency structures using the same head-finding rules. To compare with their results, we run our baseline and QG-enhanced O2 parsers on CTB5X. Table 9 presents the results.<sup>11</sup> The indirect comparison indicates that our approach can achieve larger improvement than their treebank conversion based method.

## 6 Conclusions

The current paper proposes a simple and effective framework for exploiting multiple large-scale treebanks of different annotation styles. We design rich TPs to model the annotation inconsistencies and consequently propose QG features based on these TPs. Extensive experiments show that our approach can effectively utilize the syntactic knowledge from another treebank and significantly improve the state-of-the-art parsing accuracy.

<sup>11</sup>We thank the authors for sharing their results. Niu et al. (2009) also use the reranker (RP) of Charniak and Johnson (2005) as a stronger baseline, but the results are missing. They find a less improvement on F score with RP than with GP (0.9% vs. 1.1%). We refer to their Table 5 and 6 for details.

## Acknowledgments

This work was supported by National Natural Science Foundation of China (NSFC) via grant 61133012, the National “863” Major Projects via grant 2011AA01A207, and the National “863” Leading Technology Research Project via grant 2012AA011102.

## References

- Mohit Bansal and Dan Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 693–702, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Bernd Bohnet. 2009. Efficient parsing of syntactic and semantic dependency structures. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 67–72, Boulder, Colorado, June. Association for Computational Linguistics.
- David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 877–886, Honolulu, Hawaii, October. Association for Computational Linguistics.
- David Burkett, Slav Petrov, John Blitzer, and Dan Klein. 2010. Learning better monolingual models with unannotated bilingual text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, CoNLL ’10, pages 46–54, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL-05*, pages 173–180.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *ANLP’00*, pages 132–139.
- Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin, and Ting Liu. 2009. Multilingual dependency-based syntactic and semantic parsing. In *Proceedings of CoNLL 2009: Shared Task*, pages 49–54.
- Keh-Jiann Chen, Chi-Ching Luo, Ming-Chung Chang, Feng-Yi Chen, Chao-Jan Chen, Chu-Ren Huang, and Zhao-Ming Gao, 2003. *Sinica treebank: Design criteria, representational issues and implementation*, chapter 13, pages 231–248. Kluwer Academic Publishers.
- Wenliang Chen, Jun’ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 570–579, Singapore, August. Association for Computational Linguistics.
- Wenliang Chen, Jun’ichi Kazama, and Kentaro Torisawa. 2010. Bitext dependency parsing with bilingual subtree constraints. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 21–29, Uppsala, Sweden, July. Association for Computational Linguistics.
- Micheal Collins, Lance Ramshaw, Jan Hajic, and Christoph Tillmann. 1999. A statistical parser for czech. In *ACL 1999*, pages 505–512.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP 2002*.
- Andrea Gesmundo, James Henderson, Paola Merlo, and Ivan Titov. 2009. A latent variable model of synchronous syntactic-semantic parsing for multiple languages. In *Proceedings of CoNLL 2009: Shared Task*, pages 37–42.
- Kevin Gimpel and Noah A. Smith. 2011. Quasi-synchronous phrase dependency grammars for machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 474–485, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Jan Hajič, Massimiliano Ciaramita, Richard Johanson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL 2009*.
- Liang Huang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1222–1231, Singapore, August. Association for Computational Linguistics.
- Wenbin Jiang, Liang Huang, and Qun Liu. 2009. Automatic adaptation of annotation standards: Chinese word segmentation and pos tagging – a case study. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 522–530, Suntec, Singapore, August. Association for Computational Linguistics.
- Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1–11, Uppsala, Sweden, July. Association for Computational Linguistics.

- Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio, June. Association for Computational Linguistics.
- Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Li. 2011. Joint models for chinese pos tagging and dependency parsing. In *EMNLP 2011*, pages 1180–1191.
- Ting Liu, Jinshan Ma, and Sheng Li. 2006. Building a dependency treebank for improving Chinese parser. In *Journal of Chinese Language and Computing*, volume 16, pages 207–224.
- Andr— F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. 2008. Stacking dependency parsers. In *EMNLP’08*, pages 157–166.
- Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL 2006*.
- Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL 2005*, pages 91–98.
- Zheng-Yu Niu, Haifeng Wang, and Hua Wu. 2009. Exploiting heterogeneous treebanks for parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 46–54, Suntec, Singapore, August. Association for Computational Linguistics.
- Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL 2008*, pages 950–958.
- Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.
- Eric W. Noreen. 1989. *Computer-intensive methods for testing hypotheses: An introduction*. John Wiley & Sons, Inc., New York. Book (ISBN 0471611360).
- Zhou Qiang. 2004. Annotation scheme for chinese treebank. *Journal of Chinese Information Processing*, 18(4):1–8.
- David Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 23–30, New York City, June. Association for Computational Linguistics.
- David A. Smith and Jason Eisner. 2009. Parser adaptation and projection with quasi-synchronous grammar features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 822–831, Singapore, August. Association for Computational Linguistics.
- Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32, Prague, Czech Republic, June. Association for Computational Linguistics.
- Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 409–420, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Fei Xia, Rajesh Bhatt, Owen Rambow, Martha Palmer, and Dipti Misra. Sharma. 2008. Towards a multi-representational treebank. In *In Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories*.
- Nianwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese Treebank: Phrase structure annotation of a large corpus. In *Natural Language Engineering*, volume 11, pages 207–238.
- Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of IWPT 2003*, pages 195–206.
- Yue Zhang and Stephen Clark. 2008a. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, pages 888–896.
- Yue Zhang and Stephen Clark. 2008b. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Guangyou Zhou, Jun Zhao, Kang Liu, and Li Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1556–1565, Portland, Oregon, USA, June. Association for Computational Linguistics.



# A Probabilistic Model for Canonicalizing Named Entity Mentions

Dani Yogatama Yanchuan Sim Noah A. Smith

Language Technologies Institute

Carnegie Mellon University

Pittsburgh, PA 15213, USA

{dyogatama, ysim, nasmith}@cs.cmu.edu

## Abstract

We present a statistical model for canonicalizing named entity mentions into a table whose rows represent entities and whose columns are attributes (or parts of attributes). The model is novel in that it incorporates entity context, surface features, first-order dependencies among attribute-parts, and a notion of noise. Transductive learning from a few seeds and a collection of mention tokens combines Bayesian inference and conditional estimation. We evaluate our model and its components on two datasets collected from political blogs and sports news, finding that it outperforms a simple agglomerative clustering approach and previous work.

## 1 Introduction

Proper handling of mentions in text of real-world entities—identifying and resolving them—is a central part of many NLP applications. We seek an algorithm that infers a set of real-world entities from mentions in a text, mapping each entity mention token to an entity, and discovers general categories of words used in names (e.g., titles and last names). Here, we use a probabilistic model to infer a structured representation of canonical forms of entity attributes through transductive learning from named entity mentions with a small number of seeds (see Table 1). The input is a collection of mentions found by a named entity recognizer, along with their contexts, and, following Eisenstein et al. (2011), the output is a table in which entities are rows (the number of which is not pre-specified) and attribute words are organized into columns.

This paper contributes a model that builds on the approach of Eisenstein et al. (2011), but also:

- incorporates context of the mention to help with disambiguation and to allow mentions that do not share words to be merged liberally;
- conditions against shape features, which improve the assignment of words to columns;

- is designed to explicitly handle some noise; and
- is learned using elements of Bayesian inference with conditional estimation (see §2).

We experiment with variations of our model, comparing it to a baseline clustering method and the model of Eisenstein et al. (2011), on two datasets, demonstrating improved performance over both at recovering a gold standard table. In a political blogs dataset, the mentions refer to political figures in the United States (e.g., Mrs. Obama and Michelle Obama). As a result, the model discovers parts of names—(Mrs., Michelle, Obama)—while simultaneously performing coreference resolution for named entity mentions. In the sports news dataset, the model is provided with named entity mentions of heterogeneous types, and success here consists of identifying the correct team for every player (e.g., Kobe Bryant and Los Angeles Lakers). In this scenario, given a few seed examples, the model begins to identify simple relations among named entities (in addition to discovering attribute structures), since attributes are expressed as named entities across multiple mentions. We believe this adaptability is important, as the salience of different kinds of names and their usages vary considerably across domains.

Bill	Clinton	Mr.			
George	Bush	Mr.			W.
Barack	Obama		Sen.		Hussein
Hillary	Clinton	Mrs.	Sen.		
Bristol	Palin	Ms.			
Emil	Jones			Jr.	
Kay	Hutchison				Bailey

Ben	Roethlisberger			Steelers
Derek	Bryant	Los	Angeles	
	Jeter	New	York	

Table 1: Seeds for politics (above) and sports (below).

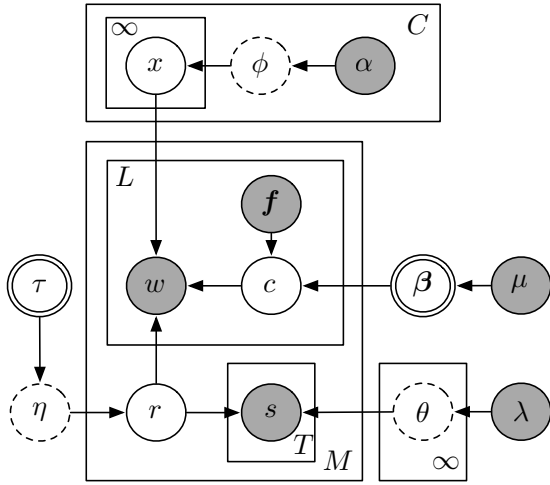


Figure 1: Graphical representation of our model. Top, the generation of the table:  $C$  is the number of attributes/columns, the number of rows is infinite,  $\alpha$  is a vector of concentration parameters,  $\phi$  is a multinomial distribution over strings, and  $x$  is a word in a table cell. Lower left, for choosing entities to be mentioned:  $\tau$  determines the stick lengths and  $\eta$  is the distribution over entities to be selected for mention. Middle right, for choosing attributes to use in a mention:  $f$  is the feature vector, and  $\beta$  is the weight vector drawn from a Laplace distribution with mean zero and variance  $\mu$ . Center, for generating mentions:  $M$  is the number of mentions in the data,  $w$  is a word token set from an entity/row  $r$  and attribute/column  $c$ . Lower right, for generating contexts:  $s$  is a context word, drawn from a multinomial distribution  $\theta$  with a Dirichlet prior  $\lambda$ . Variables that are known or fixed are shaded; variables that are optimized are double circled. Others are latent; dashed lines imply collapsing.

## 2 Model

We begin by assuming as input a set of mention tokens, each one or more words. In our experiments these are obtained by running a named entity recognizer. The output is a table in which rows are understood to correspond to entities (types, not mention tokens) and columns are fields, each associated with an attribute or a part of it. Our approach is based on a probabilistic graphical model that generates the mentions, which are observed, and the table, which is mostly unobserved, similar to Eisenstein et al. (2011). Our learning procedure is a hybrid of Bayesian inference and conditional estimation. The generative story, depicted in Figure 1, is:

- For each column  $j \in \{1, \dots, C\}$ :
  - Draw a multinomial distribution  $\phi_j$  over the

- vocabulary from a Dirichlet process:  $\phi_j \sim \text{DP}(\alpha_j, G_0)$ . This is the lexicon for field  $j$ .
- Generate table entries. For each row  $i$  (of which there are infinitely many), draw an entry  $x_{i,j}$  for cell  $i, j$  from  $\phi_j$ . A few of these entries (the seeds) are observed; we denote those  $\tilde{x}$ .
- Draw weights  $\beta_j$  that associate shape and positional features with columns from a 0-mean,  $\mu$ -variance Laplace distribution.
- Generate the distribution over entities to be mentioned in general text:  $\eta \sim \text{GEM}(\tau)$  (“stick-breaking” distribution).
- Generate context distributions. For each row  $r$ :
  - Draw a multinomial over the context vocabulary (distinct from mention vocabulary) from a Dirichlet distribution,  $\theta_r \sim \text{Dir}(\lambda)$ .
- For each mention token  $m$ :
  - Draw an entity/row  $r \sim \eta$ .
  - For each word in the mention  $w$ , given some of its features  $f$  (assumed observed):
    - ▷ Choose a column  $c \sim \frac{1}{Z} \exp(\beta_c^\top f)$ . This uses a log-linear distribution with partition function  $Z$ . In one variation of our model, first-order dependencies among the columns are enabled; these introduce a dynamic character to the graphical model that is not shown in Figure 1.
    - ▷ With probability  $1 - \epsilon$ , set the text  $w_{ml}$  to be  $x_{rc}$ . Otherwise, generate any word from a unigram-noise distribution.
  - Generate mention context. For each of the  $T = 10$  context positions (five before and five after the mention), draw the word  $s$  from  $\theta_r$ .

Our choices of prior distributions reflect our beliefs about the shapes of the various distributions. We expect field lexicons  $\phi_j$  and the distributions over mentioned entities  $\eta$  to be “Zipfian” and so use tools from nonparametric statistics to model them. We expect column-feature weights  $\beta$  to be mostly zero, so a sparsity-inducing Laplace prior is used (Tibshirani, 1996).

Our goal is to maximize the *conditional* likelihood of most of the evidence (mentions, contexts, and seeds),  $p(w, s, \tilde{x} \mid \alpha, \beta, \lambda, \tau, \mu, \epsilon, f) =$

$$\sum_r \sum_c \sum_{x \setminus \tilde{x}} \int d\theta \int d\eta \int d\phi p(w, s, r, c, x, \theta, \eta, \phi \mid \alpha, \beta, \lambda, \tau, \mu, \epsilon, f)$$

with respect to  $\beta$  and  $\tau$ . We fix  $\alpha$  (see §3.3 for the values of  $\alpha$  for each dataset),  $\lambda = 2$  (equivalent to add-one smoothing),  $\mu = 2 \times 10^{-8}$ ,  $\epsilon = 10^{-10}$ , and each mention word’s  $f$ . Fixing  $\lambda$ ,  $\mu$ , and  $\alpha$  is essentially just “being Bayesian,” or fixing a hyperparameter based on prior beliefs. Fixing  $f$  is quite different; it is conditioning our model on some observable features of the data, in this case word shape features. We do this to avoid integrating over feature vector values. These choices highlight that the design of a probabilistic model can draw from both Bayesian and discriminative tools. Observing some of  $x$  as seeds ( $\tilde{x}$ ) renders this approach transductive.

Exact inference in this model is intractable, so we resort to an approximate inference technique based on Markov Chain Monte Carlo simulation. The optimization of  $\beta$  can be described as “contrastive” estimation (Smith and Eisner, 2005), in which some aspects of the data are conditioned against for computational convenience. The optimization of  $\tau$  can be described as “empirical Bayesian” estimation (Morris, 1983) in which the parameters of a prior are fit to data. Our overall learning procedure is a Monte Carlo Expectation Maximization algorithm (Wei and Tanner, 1990).

### 3 Learning and Inference

Our learning procedure is an iterative algorithm consisting of two steps. In the E-step, we perform collapsed Gibbs sampling to obtain distributions over row and column indices for every mention, given the current value of the hyperparameters. In the M-step, we obtain estimates for the hyperparameters, given the current posterior distributions.

#### 3.1 E-step

For the  $m$ th mention, we sample row index  $r$ , then for each word  $w_{m\ell}$ , we sample column index  $c$ .

##### 3.1.1 Sampling Rows

Similar to Eisenstein et al. (2011), when we sample the row for a mention, we use Bayes’ rule and marginalize the columns. We further incorporate context information and a notion of noise.

$$p(r_m = r \mid \dots) \propto p(r_m = r \mid r_{-m}, \eta) \left( \prod_{\ell} \sum_c p(w_{m\ell} \mid x, r_m = r, c_{m\ell} = c) \right) \left( \prod_t p(s_{mt} \mid r_m = r) \right)$$

We consider each quantity in turn.

**Prior.** The probability of drawing a row index follows a stick breaking distribution. This allows us to have an unbounded number of rows and let the model infer the optimal value from data. A standard marginalization of  $\eta$  gives us:

$$p(r_m = r \mid r_{-m}, \tau) = \begin{cases} \frac{N_r^{-m}}{N + \tau} & \text{if } N_r^{-m} > 0 \\ \frac{\tau}{N + \tau} & \text{otherwise,} \end{cases}$$

where  $N$  is the number of mentions,  $N_r$  is the number of mentions assigned to row  $r$ , and  $N_r^{-m}$  is the number of mentions assigned to row  $r$ , excluding  $m$ .

**Mention likelihood.** In order to compute the likelihood of observing mentions in the dataset, we have to consider a few cases. If a cell in a table has already generated a word, it can only generate that word. This hard constraint was a key factor in the inference algorithm of Eisenstein et al. (2011); we speculate that softening it may reduce MCMC mixing time, so introduce a notion of noise. With probability  $\epsilon = 10^{-10}$ , the cell can generate any word. If a cell has not generated any word, its probability still depends on other elements of the table. With base distribution  $G_0$ ,<sup>1</sup> and marginalizing  $\phi$ , we have:

$$p(w_{m\ell} \mid x, r_m = r, c_{m\ell} = c, \alpha_c) = \begin{cases} 1 - \epsilon & \text{if } x_{rc} = w_{m\ell} \\ \epsilon & \text{if } x_{rc} \notin \{w_{m\ell}, \emptyset\} \\ \frac{N_{cw}^{-m\ell}}{N_c^{-m\ell} + \alpha_c} & \text{if } x_{rc} = \emptyset \text{ and } N_{cw} > 0 \\ G_0(w_{m\ell}) \frac{\alpha_c}{N_c^{-m\ell} + \alpha_c} & \text{if } x_{rc} = \emptyset \text{ and } N_{cw} = 0 \end{cases} \quad (1)$$

where  $N_c^{-m\ell}$  is the number of cells in column  $c$  that are not empty and  $N_{cw}^{-m\ell}$  is the number of cells in column  $c$  that are set to the word  $w_{m\ell}$ ; both counts excluding the current word under consideration.

**Context likelihood.** It is important to be able to use context information to determine which row a mention should go into. As a novel extension, our model also uses surrounding words of a mention as its “context”—similar context words can encourage two mentions that do not share any words to be merged. We choose a Dirichlet-multinomial distribution for our context distribution. For every row in the table, we have a multinomial distribution over context vocabulary  $\theta_r$  from a Dirichlet prior  $\lambda$ .

<sup>1</sup>We let  $G_0$  be a uniform distribution over the vocabulary.

Therefore, the probability of observing the  $t$ th context word for mention  $m$  is  $p(s_{mt} | r_m = r, \lambda)$

$$= \begin{cases} \frac{N_r^{-mt} + \lambda_s - 1}{N_r^{-mt} + \sum_v \lambda_v - V} & \text{if } N_r^{-mt} > 0 \\ \frac{\lambda_s - 1}{\sum_v \lambda_v - V} & \text{otherwise,} \end{cases}$$

where  $N_r^{-mt}$  is the number of context words of mentions assigned to row  $r$ ,  $N_{rs}^{-mt}$  is the number of context words of mentions assigned to row  $r$  that are  $s_{mt}$ , both excluding the current context word, and  $v$  ranges over the context vocabulary of size  $V$ .

### 3.1.2 Sampling Columns

Our column sampling procedure is novel to this work and substantially differs from that of Eisenstein et al. (2011). First, we note that when we sample column indices for each word in a mention, the row index for the mention  $r$  has already been sampled. Also, our model has interdependencies among column indices of a mention.<sup>2</sup> Standard Gibbs sampling procedure breaks down these dependencies. For faster mixing, we experiment with first-order dependencies between columns when sampling column indices. This idea was suggested by Eisenstein et al. (2011, footnote 1) as a way to learn structure in name conventions. We suppressed this aspect of the model in Figure 1 for clarity.

We sample the column index  $c_1$  for the first word in the mention, marginalizing out probabilities of other words in the mention. After we sample the column index for the first word, we sample the column index  $c_2$  for the second word, fixing the previous word to be in column  $c_1$ , and marginalizing out probabilities of  $c_3, \dots, c_L$  as before. We repeat the above procedure until we reach the last word in the mention. In practice, this can be done efficiently using backward probabilities computed via dynamic programming. This kind of blocked Gibbs sampling was proposed by Jensen et al. (1995) and used in NLP by Mochihashi et al. (2009). We have:

$$p(c_{ml} = c | \mathbf{f}_{ml}, \beta) p(c_{ml} = c | c_{ml_-} = c_-) \left( \sum_{c_+} p_b(c_{ml} = c | c_{ml_+} = c_+) \right) p(w_{ml} | x, r_m = r, c_{ml} = c, \alpha_c),$$

<sup>2</sup>As shown in Figure 1, column indices in a mention form “v-structures” with the row index  $r$ . Since every  $w_\ell$  is observed, there is an active path that goes through all these nodes.

where  $\ell_-$  is the preceding word and  $c_-$  is its sampled index,  $\ell_+$  is the following word and  $c_+$  is its possible index, and  $p_b(\cdot)$  are backward probabilities. Alternatively, we can perform standard Gibbs sampling and drop the dependencies between columns, which makes the model rely more heavily on the features. For completeness, we detail the computations.

**Featurized log linear distribution.** Our model can use arbitrary features to choose a column index. These features are incorporated as a log-linear distribution,  $p(c_{ml} = c | \mathbf{f}_{ml}, \beta) = \frac{\exp(\beta_c^\top \mathbf{f}_{ml})}{\sum_{c'} \exp(\beta_{c'}^\top \mathbf{f}_{ml})}$ . The list of features used in our experiments is:  $\mathbf{1}\{w$  is the first word in the mention};  $\mathbf{1}\{w$  ends with a period};  $\mathbf{1}\{w$  is the last word in the mention};  $\mathbf{1}\{w$  is a Roman numeral};  $\mathbf{1}\{w$  starts with an upper-case letter};  $\mathbf{1}\{w$  is an Arabic number};  $\mathbf{1}\{w \in \{\text{mr, mrs, ms, miss, dr, mdm}\}$ };  $\mathbf{1}\{w$  contains  $\geq 1$  punctuation symbol};  $\mathbf{1}\{w \in \{\text{jr, sr}\}$ };  $\mathbf{1}\{w \in \{\text{is, in, of, for}\}$ };  $\mathbf{1}\{w$  is a person entity};  $\mathbf{1}\{w$  is an organization entity}.

**Forward and backward probabilities.** Since we introduce first-order dependencies between columns, we have forward and backward probabilities, as in HMMs. However, we always sample from left to right, so we do not need to marginalize random variables to the left of the current variable because their values are already sampled. Our transition probabilities are as follows:

$$p(c_{ml} = c | c_{ml_-} = c_-) = \frac{N_{c_-,c}^{-m}}{\sum_{c'_-} N_{c'_-,c}^{-m}},$$

where  $N_{c_-,c}^{-m}$  is the number of times we observe transitions from column  $c_-$  to  $c$ , excluding mention  $m$ . The forward and backward equations are simple (we omit them for space).

**Mention likelihood.** Mention likelihood  $p(w_{ml} | x, r_m = r, c_{ml} = c, \alpha_c)$  is identical to when we sample the row index (Eq. 1).

## 3.2 M-step

In the M-step, we use gradient-based optimization routines, L-BFGS (Liu and Nocedal, 1989) and OWL-QN (Andrew and Gao, 2007) respectively, to maximize with respect to  $\tau$  and  $\beta$ .

### 3.3 Implementation Details

We ran Gibbs sampling for 500 iterations,<sup>3</sup> discarding the first 200 for burn-in and averaging counts over every 10th sample to reduce autocorrelation.

For each word in a mention  $w$ , we introduced 12 binary features  $f$  for our featurized log-linear distribution (§3.1.2).

We then downcased all words in mentions for the purpose of defining the table and the mention words  $w$ . Ten context words (5 each to the left and right) define  $s$  for each mention token.

For non-convex optimization problems like ours, initialization is important. To guide the model to reach a good local optimum without many restarts, we manually initialized feature weights and put a prior on transition probabilities to reflect phenomena observed in the initial seeds. The initializer was constructed once and not tuned across experiments.<sup>4</sup> The M-step was performed every 50 Gibbs sampling iterations. After inference, we filled each cell with the word that occurred at least 80% of the time in the averaged counts for the cell, if such a word existed.

## 4 Experiments

We compare several variations of our model to Eisenstein et al. (2011) (the authors provided their implementation to us) and a clustering baseline.

### 4.1 Datasets

We collected named entity mentions from two corpora: political blogs and sports news. The political blogs corpus is a collection of blog posts about politics in the United States (Eisenstein and Xing, 2010), and the sports news corpus contains news summaries of major league sports games (National Basketball

<sup>3</sup>On our moderate-sized datasets (see §4.1), each iteration takes approximately three minutes on a 2.2GHz CPU.

<sup>4</sup>For the politics dataset, we set  $C = 6$ ,  $\alpha = \langle 1.0, 1.0, 10^{-12}, 10^{-15}, 10^{-12}, 10^{-8} \rangle$ , initialized  $\tau = 1$ , and used a Dirichlet prior on transition counts such that before observing any data:  $N_{0,1} = 10, N_{0,5} = 5, N_{2,0} = 10, N_{2,1} = 10, N_{2,3} = 10, N_{2,4} = 5, N_{3,0} = 10, N_{3,1} = 10, N_{5,1} = 15$  (others are set to zero). For the sports dataset, we set  $C = 5$ ,  $\alpha = \langle 1.0, 1.0, 10^{-15}, 10^{-6}, 10^{-6} \rangle$ , initialized  $\tau = 1$ , and used a Dirichlet prior on transition counts  $N_{0,1} = 10, N_{2,3} = 20, N_{3,4} = 10$  (others are set to zero). We also manually initialized the weights of some features  $\beta$  for both datasets. These values were obtained from preliminary experiments on a smaller sample of the datasets, and updated on the first EM iteration.

	Politics	Sports
# source documents	3,000	700
# mentions	10,647	13,813
# unique mentions	528	884
size of mention vocabulary	666	1,177
size of context vocabulary	2,934	2,844

Table 2: Descriptive statistics about the datasets.

Association, National Football League, and Major League Baseball) in 2009. Due to the large size of the corpora, we uniformly sampled a subset of documents for each corpus and ran the Stanford NER tagger (Finkel et al., 2005), which tagged named entities mentions as person, location, and organization. We used named entity of type person from the political blogs corpus, while we are interested in person and organization entities for the sports news corpus. Mentions that appear less than five times are discarded. Table 2 summarizes statistics for both datasets of named entity mentions.

**Reference tables.** We use Eisenstein et al.’s manually built 125-entity (282 vocabulary items) reference table for the politics dataset. Each entity in the table is represented by the set of all tokens that appear in its references, and the tokens are placed in its correct column. For the sports data, we obtained a roster of all NBA, NFL, and MLB players in 2009. We built our sports reference table using the players’ names, teams and locations, to get 3,642 players and 15,932 vocabulary items. The gold standard table for the politics dataset is incomplete, whereas it is complete for the sports dataset.

**Seeds.** Table 1 shows the seeds for both datasets.

### 4.2 Evaluation Scores

We propose both a row evaluation to determine how well a model disambiguates entities and merges mentions of the same entity and a column evaluation to measure how well the model relates words used in different mentions. Both scores are new for this task.

The first step in evaluation is to find a maximum score bipartite matching between rows in the response and reference table.<sup>5</sup> Given the response and

<sup>5</sup>Treating each row as a set of words, we can optimize the matching using the Jonker and Volgenant (1987) algorithm. The column evaluation is identical, except that sets of words that are matched are defined by columns. We use the Jaccard similarity—for two sets  $A$  and  $B$ ,  $\frac{|A \cap B|}{|A \cup B|}$ —for our similarity function,  $\text{Sim}(i, j)$ .

reference tables,  $x_{res}$  and  $x_{ref}$ , we can compute:

$$S_{res} = \frac{1}{|x_{res}|} \sum_{i \in x_{res}, j \in x_{ref}: \text{Match}(i, j) = 1} \text{Sim}(i, j)$$

$$S_{ref} = \frac{1}{|x_{ref}|} \sum_{i \in x_{res}, j \in x_{ref}: \text{Match}(i, j) = 1} \text{Sim}(i, j)$$

where  $i$  and  $j$  denote rows,  $\text{Match}(i, j)$  is one if  $i$  and  $j$  are matched to each other in the optimal matching or zero otherwise.  $S_{res}$  is a precision-like score, and  $S_{ref}$  is a recall-like score.<sup>6</sup> Column evaluation is the same, but compares columns instead.

### 4.3 Baselines

Our simple baseline is an agglomerative clustering algorithm that clusters mentions into entities using the single-linkage criterion. Initially, each unique mention forms its own cluster that we incrementally merge together to form rows in the table. This method requires a similarity score between two clusters. For the politics dataset, we follow Eisenstein et al. (2011) and use the string edit distance between mention strings in each cluster to define the score. For the sports dataset, since mentions contain person and organization named entity types, our score for clustering uses the Jaccard distance between context words of the mentions. However, such clusterings do not produce columns. Therefore, we first match words in mentions of type person against a regular expression to recognize entity attributes from a fixed set of titles and suffixes, and the first, middle and last names. We treat words in mentions of type organization as a single attribute.<sup>7</sup> As we merge clusters together, we arrange words such that

<sup>6</sup>Eisenstein et al. (2011) used precision and recall for their similarity function. Precision prefers a more compact response row (or column), which unfairly penalizes situations like those of our sports dataset, where rows are heterogeneous (e.g., including people and organizations). Consider a response table made up of two rows: (Kobe, Bryant) and (Los, Angeles, Lakers), and a reference table containing all NBA players and their team names, e.g., (Kobe, Bryant, Los, Angeles, Lakers). Evaluating with the precision similarity function, we will have perfect precision by matching the first row to the reference row for Kobe Bryant and the latter row to any Lakers player. The system is not rewarded for merging the two rows together, even if they are describing the same entity. Our evaluation scores, however, reward the system for accurately filling in more cells.

<sup>7</sup>Note that the baseline system uses NER tags, list of titles and suffixes; which are also provided to our model through the features in §3.1.2.

all words within a column belong to the same attribute, creating columns as necessary to accommodate multiple similar attributes as a result of merging. We can evaluate the tables produced by each step of the clustering to obtain the entire sequence of response-reference scores.

As a strong baseline, we also compare our approach with the original implementation of the model of Eisenstein et al. (2011), denoted by EEA.

### 4.4 Results

For both the politics and sports dataset, we run the procedure in §3.3 three times and report the results.

**Politics.** The results for the politics dataset are shown in Figure 2. Our model consistently outperformed both baselines. We also analyze how much each of our four main extensions (shape features, context information, noise, and first-order column dependencies) to EEA contributes to overall performance by ablating each in turn (also shown in Fig. 2). The best-performing complete model has 415 rows, of which 113 were matched to the reference table. Shape features are useful: removing them was detrimental, and they work even better without column dependencies. Indeed, the best model did not have column dependencies. Removing context features had a strong negative effect, though perhaps this could be overcome with a more carefully tuned initializer.

In row evaluation, the baseline system can achieve a high reference score by creating one entity row per unique string, but as it merges strings, the clusters encompass more word tokens, improving response score at the expense of reference score. With fewer clusters, there are fewer entities in the response table for matching and the response score suffers. Although we use the same seed table in both experiments, the results from EEA are below the baseline curve because it has the additional complexity of inferring the number of columns from data. Our model is simpler in this regard since it assumes that the number of columns is known ( $C = 6$ ). In column evaluation, our method without column dependencies was best.

**Sports.** The results for the sports dataset are shown in Figure 3. Our best-performing complete model’s response table contains 599 rows, of which 561 were matched to the reference table. In row eval-

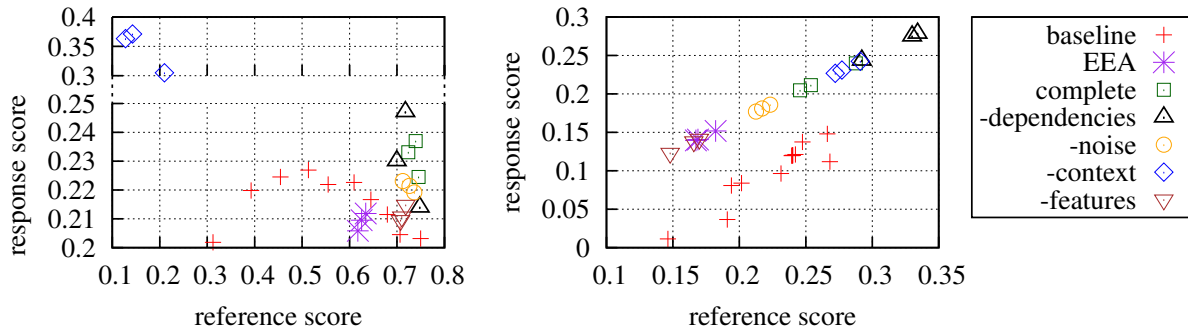


Figure 2: Row (left) and column (right) scores for the politics dataset. For all but “baseline” (clustering), each point denotes a unique sampling run. Note the change in scale in the left plot at  $y = 0.25$ . For the clustering baseline, points correspond to iterations.

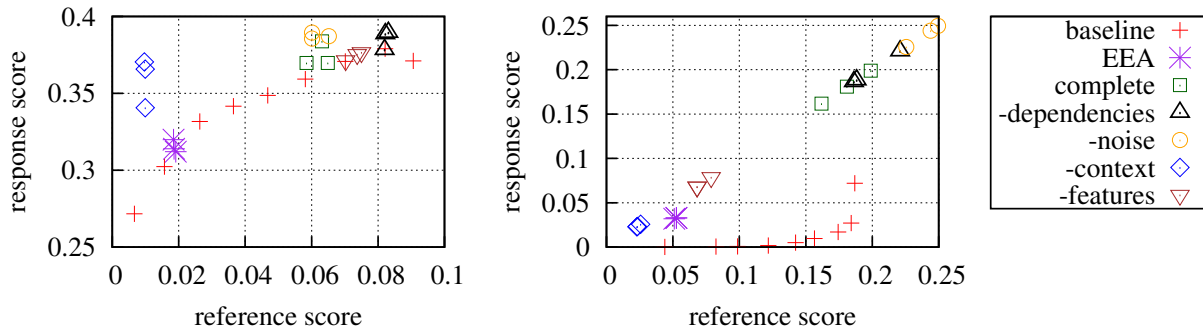


Figure 3: Row (left) and column (right) scores for the sports dataset. Each point denotes a unique sampling run. The reference score is low since the reference set includes all entities in the NBA, NFL, and MLB, but most of them were not mentioned in our dataset.

uation, our model lies above the baseline response-reference score curve, demonstrating its ability to correctly identify and combine player mentions with their team names. Similar to the previous dataset, our model is also substantially better in column evaluation, indicating that it mapped mention words into a coherent set of five columns.

#### 4.5 Discussion

The two datasets illustrate that our model adapts to somewhat different tasks, depending on its input. Furthermore, fixing  $C$  (unlike EEA) does appear to have benefits.

In the politics dataset, most of the mentions contain information about people. Therefore, besides canonicalizing named entities, the model also resolves within-document and cross-document coreference, since it assigned a row index for every mention. For example, our model learned that most mentions of John McCain, Sen. John McCain, Sen. McCain, and Mr. McCain refer to the same entity. Table 3 shows a few noteworthy entities from our complete model’s output table.

Barack	Obama	Mr.	Sen.		Hussein
Michelle	Obama	Mrs.			
Norm	Coleman		Sen.		
Sarah	Palin	Ms.			
John	McCain	Mr.	Sen.		Hussein

Table 3: A small segment of the output table for the politics dataset, showing a few noteworthy correct (blue) and incorrect (red) examples. Black indicates seeds. Though Ms. is technically correct, there is variation in preferences and conventions. Our data include eight instances of Ms. Palin and none of Mrs. Palin or Mrs. Sarah Palin.

The first entity is an easy example since it only had to complete information provided in the seed table. The model found the correct gender-specific title for Barack Obama, Mr.. The rest of the examples were fully inferred from the data. The model was essentially correct for the second, third, and fourth entities. The last row illustrates a partially erroneous example, in which the model confused the middle name of John McCain, possibly because of a combination of a strong prior to reuse this row and the

Derek	Jeter	New	York	
Ben	Roethlisberger		Pittsburgh	Steelers
Alex	Rodriguez	New	York	Yankees
Michael	Vick		Philadelphia	Eagles
Kevin	Garnett	Los	Angeles	Lakers
Dave	Toub		The	Bears

Table 4: A small segment of the output table for the sports dataset, showing a few noteworthy correct (blue) and incorrect (red) examples. Black indicates seed examples.

introduction of a notion of noise.

In the sports dataset, persons and organizations are mentioned. Recall that success here consists of identifying the correct team for every player. The EEA model is not designed for this and performed poorly. Our model can do better, since it makes use of context information and features, and it can put a person and an organization in one row even though they do not share common words. Table 4 shows a few noteworthy entities from our complete model’s output.

Surprisingly, the model failed to infer that Derek Jeter plays for New York Yankees, although mentions of the organization New York Yankees can be found in our dataset. This is because the model did not see enough evidence to put them in the same row. However, it successfully inferred the missing information for Ben Roethlisberger. The next two rows show cases where our model successfully matched players with their teams and put each word token to its respective column. The most frequent error, by far, is illustrated in the fifth row, where a player is matched with a wrong team. Kevin Garnett plays for the Boston Celtics, not the Los Angeles Lakers. It shows that in some cases context information is not adequate, and a possible improvement might be obtained by providing more context to the model. The sixth row is interesting because Dave Toub is indeed affiliated with the Chicago Bears. However, when the model saw a mention token The Bears, it did not have any other columns to put the word token The, and decided to put it in the fourth column although it is not a location. If we added more columns, determiners could become another attribute of the entities that might go into one of these new columns.

## 5 Related Work

There has been work that attempts to fill predefined templates using Bayesian nonparametrics (Haghighi and Klein, 2010) and automatically learns template structures using agglomerative clustering (Chambers and Jurafsky, 2011). Charniak (2001) and El-sner et al. (2009) focused specifically on names and discovering their structure, which is a part of the problem we consider here. More similar to our work, Eisenstein et al. (2011) introduced a non-parametric Bayesian approach to extract structured databases of entities. A fundamental difference of our approach from any of the previous work is it maximizes conditional likelihood and thus allows beneficial incorporation of arbitrary features.

Our model is focused on the problem of canonicalizing mention strings into their parts, though its  $r$  variables (which map mentions to rows) could be interpreted as (within-document and cross-document) coreference resolution, which has been tackled using a range of probabilistic models (Li et al., 2004; Haghighi and Klein, 2007; Poon and Domingos, 2008; Singh et al., 2011). We have not evaluated it as such, believing that further work should be done to integrate appropriate linguistic cues before such an application.

## 6 Conclusions

We presented an improved probabilistic model for canonicalizing named entities into a table. We showed that the model adapts to different tasks depending on its input and seeds, and that it improves over state-of-the-art performance on two corpora.

## Acknowledgements

The authors thank Jacob Eisenstein and Tae Yano for helpful discussions and providing us with the implementation of their model, Tim Hawes for helpful discussions, Naomi Saphra for assistance in developing the gold standard for the politics dataset, and three anonymous reviewers for comments on an earlier draft of this paper. This research was supported in part by the U.S. Army Research Office, Google’s sponsorship of the Worldly Knowledge project at CMU, and A\*STAR (fellowship to Y. Sim); the contents of this paper do not necessarily reflect the position or the policy of the sponsors, and no official endorsement should be inferred.



## References

- G. Andrew and J. Gao. 2007. Scalable training of L1-regularized log-linear models. In *Proc. of ICML*.
- N. Chambers and D. Jurafsky. 2011. Template-based information extraction without the templates. In *Proc. of ACL-HLT*.
- E. Charniak. 2001. Unsupervised learning of name structure from coreference data. In *Proc. of NAACL*.
- J. Eisenstein and E. P. Xing. 2010. The CMU 2008 political blog corpus. Technical report, Carnegie Mellon University.
- J. Eisenstein, T. Yano, W. W. Cohen, N. A. Smith, and E. P. Xing. 2011. Structured databases of named entities from Bayesian nonparametrics. In *Proc. of EMNLP Workshop on Unsupervised Learning in NLP*.
- M. Elsner, E. Charniak, and M. Johnson. 2009. Structured generative models for unsupervised named-entity clustering. In *Proc. of NAACL-HLT*.
- J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proc. of ACL*.
- A. Haghighi and D. Klein. 2007. Unsupervised coreference resolution in a nonparametric Bayesian model. In *Proc. of ACL*.
- A. Haghighi and D. Klein. 2010. An entity-level approach to information extraction. In *Proc. of ACL Short Papers*.
- C. S. Jensen, U. Kjaerulff, and A. Kong. 1995. Blocking Gibbs sampling in very large probabilistic expert system. *International Journal of Human-Computer Studies*, 42(6):647–666.
- R. Jonker and A. Volgenant. 1987. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340.
- X. Li, P. Morie, and D. Roth. 2004. Identification and tracing of ambiguous names: discriminative and generative approaches. In *Proc. of AAAI*.
- D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming B*, 45(3):503–528.
- D. Mochihashi, T. Yamada, and N. Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proc. of ACL-IJCNLP*.
- C. Morris. 1983. Parametric empirical Bayes inference: Theory and applications. *Journal of the American Statistical Association*, 78(381):47–65.
- H. Poon and P. Domingos. 2008. Joint unsupervised coreference resolution with Markov logic. In *Proc. of EMNLP*.
- S. Singh, A. Subramanya, F. Pereira, and A. McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proc. of ACL-HLT*.
- N. A. Smith and J. Eisner. 2005. Contrastive estimation: training log-linear models on unlabeled data. In *Proc. of ACL*.
- R. Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society B*, 58(1):267–288.
- G. C. G. Wei and M. A. Tanner. 1990. A Monte Carlo implementation of the EM algorithm and the poor man’s data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704.

# Multilingual Named Entity Recognition using Parallel Data and Metadata from Wikipedia

Sungchul Kim\*

POSTECH

Pohang, South Korea

subright@postech.ac.kr

Kristina Toutanova

Microsoft Research

Redmond, WA 98502

kristout@microsoft.com

Hwanjo Yu

POSTECH

Pohang, South Korea

hwanjoyu@postech.ac.kr

## Abstract

In this paper we propose a method to automatically label multi-lingual data with named entity tags. We build on prior work utilizing Wikipedia metadata and show how to effectively combine the weak annotations stemming from Wikipedia metadata with information obtained through English-foreign language parallel Wikipedia sentences. The combination is achieved using a novel semi-CRF model for foreign sentence tagging in the context of a parallel English sentence. The model outperforms both standard annotation projection methods and methods based solely on Wikipedia metadata.

## 1 Introduction

Named Entity Recognition (NER) is a frequently needed technology in NLP applications. State-of-the-art statistical models for NER typically require a large amount of training data and linguistic expertise to be sufficiently accurate, which makes it nearly impossible to build high-accuracy models for a large number of languages.

Recently, there have been two lines of work which have offered hope for creating NER analyzers in many languages. The first has been to devise an algorithm to tag foreign language entities using metadata from the semi-structured Wikipedia repository: inter-wiki links, article categories, and cross-language links (Richman and Schone, 2008). The second has been to use parallel English-foreign language data, a high-quality NER tagger for English, and projected annotations for the foreign language (Yarowsky et al., 2001; Das and Petrov, 2011). Parallel data has also been used to improve existing monolingual taggers or other analyzers in two languages (Burkett et al., 2010a; Burkett et al., 2010b).

---

This research was conducted during the author's internship at Microsoft Research

The goal of this work is to create high-accuracy NER annotated data for foreign languages. Here we combine elements of both Wikipedia metadata-based approaches and projection-based approaches, making use of parallel sentences extracted from Wikipedia. We propose a statistical model which can combine the two types of information. Similarly to the joint model of Burkett et al. (2010a), our model can incorporate both monolingual and bilingual features in a log-linear framework. The advantage of our model is that it is much more efficient as it does not require summing over matchings of source and target entities. It is a conditional model for target sentence annotation given an aligned English source sentence, where the English sentence is used only as a source of features. Exact inference is performed using standard semi-markov CRF model inference techniques (Sarawagi and Cohen, 2004).

Our results show that the semi-CRF model improves on the performance of projection models by more than 10 points in F-measure, and that we can achieve tagging F-measure of over 91 using a very small number of annotated sentence pairs.

The paper is organized as follows: We first describe the datasets and task setting in Section 2. Next, we present our two baseline methods: A Wikipedia metadata-based tagger and a cross-lingual projection tagger in Sections 3 and 4, respectively. We present our direct semi-CRF tagging model in Section 5.

## 2 Data and task

As a case study, we focus on two very different foreign languages: Korean and Bulgarian. The English and foreign language sentences that comprise our training and test data are extracted from Wikipedia (<http://www.wikipedia.org>). Currently there are more than 3.8 million articles in the English Wikipedia, 125,000 in the Bulgarian Wikipedia, and 131,000 in the Korean Wikipedia.

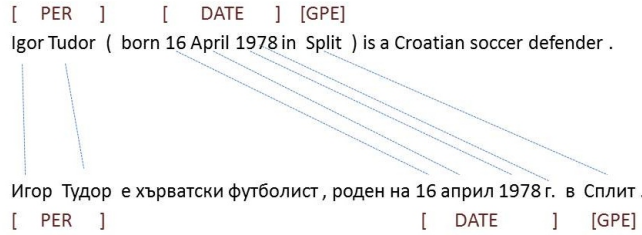


Figure 1: A parallel sentence-pair showing gold-standard NE labels and word alignments.

To create our dataset, we followed Smith et al. (2010) to find parallel-foreign sentences using comparable documents linked by inter-wiki links. The approach uses a small amount of manually annotated article-pairs to train a document-level CRF model for parallel sentence extraction. A total of 13,410 English-Bulgarian and 8,832 English-Korean sentence pairs were extracted.

Of these, we manually annotated 91 English-Bulgarian and 79 English-Korean sentence pairs with source and target named entities as well as word-alignment links among named entities in the two languages. Figure 1 illustrates a Bulgarian-English sentence pair with alignment.

The named entity annotation scheme followed has the labels GPE (Geopolitical entity), PER (Person), ORG (Organization), and DATE. It is based on the MUC-7 annotation guidelines, and GPE is synonymous with Location. The annotation process was not as rigorous as one might hope, due to lack of resources. The English-Bulgarian and English-Korean datasets were labeled by one annotator each and then annotations on the English sentences were double-checked by the other annotator. Disagreements were rare and were resolved after discussion.

The task we evaluate on is tagging of foreign language sentences. We measure performance by labeled precision, recall, and F-measure. We give partial credit if entities partially overlap on their span of words and match on their labels.

Table 1 shows the total number of English, Bulgarian and Korean entities and the percentage of entities that were manually aligned to an entity of the same type in the other language. The data sizes are fairly small as the data is

Language	Entities	Aligned %
English	342	93.9%
Bulgarian	344	93.3%
English	414	88.4%
Korean	423	86.5%

Table 1: English-Bulgarian and English-Korean data characteristics.

used only to train models with very few coarse-grained features and for evaluation. These datasets are available at <http://research.microsoft.com/en-us/people/kristout/nerwikidownload.aspx>.

As we can see, less than 100% of entities have parallels in the other language. This is due to two phenomena: one is that the parallel sentences sometimes contain different amounts of information and one language might use more detail than the other. The other is that the same information might be expressed using a named entity in one language, and using a non-entity phrase in the other language (e.g. “He is from Bulgaria” versus “He is Bulgarian”). Both of these causes of divergence are much more common in the English-Korean dataset than in the English-Bulgarian one.

### 3 Wiki-based tagger: annotating sentences based on Wikipedia metadata

We followed the approach of Richman and Schone (2008) to derive named entity annotations of both English and foreign phrases in Wikipedia, using Wikipedia metadata. The following sources of information were used from Wikipedia: *category* annotations on English documents, *article links* which link from phrases in an article to another article in the same language, and *interwiki links* which link

### English candidate entities

*Local-wiki-links:* [Igor Tudor]-PER [16 April 1978]-DATE [Croatian]-GPE

*Global-wiki-links:* [Igor Tudor]-PER\* [16]-DATE\* [16]-GPE [April]-DATE [1978]-DATE\* [1978]-GPE [Split]-GPE [Croatian]-GPE\* [Croatian]-ORG [Igor]-PER

*Stanford tagger:* [Igor Tudor]-PER [April 1978]-DATE

### Bulgarian candidate entities

*Local-wiki-links:* [Игор Тудор]-PER [16 април]-DATE

*Global-wiki-links:* : [Игор Тудор]-PER [Игор]-PER [16]-DATE [16 април]-DATE

Figure 2: Candidate NEs for the English and Bulgarian sentences according to baseline taggers.

from articles in one language to comparable (semantically equivalent) articles in the other language. In addition to the Wikipedia-derived resources, the approach requires a manually specified map from English category key-phrases to NE tags, but does not require expert knowledge for any non-English language. We implemented the main ideas of the approach but some implementation details may differ.

To tag English language phrases, we first derived named entity categorizations of English article titles, by assigning a tag based on the article’s category information. The category-to-NE map used for the assignment is a small manually specified map from phrases appearing in category titles to NE tags. For example, if an article has categories “People by”, “People from”, “Surnames” etc., it is classified as PER. Looking at the example in Figure 1, the article with title “Igor Tudor” is classified as PER because one of its categories is “Living people”. The full map we use is taken from the paper (Richman and Schone, 2008).

Using the article-level annotations and *article links* we define a local English wiki-based tagger and a global English wiki-based tagger, which will be described in detail next.

**Local English Wiki-based tagger.** This Wiki-based tagger tags phrases in an English article based on the *article links* from these phrases to NE-tagged articles. For example, suppose that the phrase “Split” in the article with title “Igor Tudor” is linked to the article with title “Split”, which is classified as GPE. Thus the local English Wiki-based tagger can tag this phrase as GPE. If, within the same article, the phrase “Split” occurs again, it can be tagged again even if it is not linked to a tagged article (this is the one sense per document assumption). Addition-

ally, the tagger tags English phrases as DATE if they match a set of manually specified regular expressions. As a filter, phrases that do not contain a capitalized word or a number are not tagged with NE tags.

**Global English Wiki-based tagger.** This tagger tags phrases with NE tags if these phrases have ever been linked to a categorized article (the most frequent label is used). For example, if “Split” does not have a link anywhere in the current article, but has been linked to the GPE-labeled article with title “Split” in another article, it will still be tagged as GPE. We also apply a local+global Wiki-tagger, which tags entities according to the local Wiki-tagger and additionally tags any non-conflicting entities according to the global tagger.

**Local foreign Wiki-based tagger.** The idea is the same as for the local English tagger, with the difference that we first assign NE tags to foreign language articles by using the NE tags assigned to English articles to which they are connected with inter-wiki links. Because we do not have maps from category phrases to NE tags for foreign languages, using inter-wiki links is a way to transfer this knowledge to the foreign languages. After we have categorized foreign language articles we follow the same algorithm as for the local English Wiki-based tagger. For Bulgarian we also filtered out entities based on capitalization and numbers, but did not do that for Korean as it has no concept of capitalization.

**Global foreign Wiki-based tagger** The global and local+global taggers are analogous, using the categorization of foreign articles as above.

Figure 2 shows the tags assigned to English and Bulgarian strings according to the local and global Wiki-based taggers. The global Wiki-based tagger could assign multiple labels to the same string (corresponding to different senses in different occurrences). In case of multiple possible labels, the most frequent one is denoted by \* in the Figure. The Figure also shows the results of the Stanford NER tagger for English (Finkel et al., 2005) (we used the MUC-7 classifier).

Table 2 reports the performance of the local (L Wiki-tagger), local+global (LG Wiki tagger) and the Stanford tagger. We can see that the local Wiki taggers have higher precision but lower recall than the local+global Wiki taggers. The local+global taggers

Language	L Wiki-tagger			LG Wiki-tagger			Stanford Tagger		
	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
English	92.8	75.1	83.0	79.7	89.5	<b>84.3</b>	86.5	77.5	81.7
Bulgarian	94.1	48.7	64.2	86.8	79.9	<b>83.2</b>			
English	92.6	75.6	83.2	84.1	86.7	<b>85.4</b>	82.2	71.9	76.7
Korean	89.5	57.3	<b>69.9</b>	43.2	78.0	55.6			

Table 2: English-Bulgarian and English-Korean Wiki-based tagger performance.

are overall best for English and Bulgarian. The local tagger is best for Korean, as the precision suffers too much due to the global tagger. This is perhaps due in part to the absence of the capitalization filter for Korean which improved precision for Bulgarian and English. The Stanford tagger is worse than the Wiki-based tagger, but it is different enough that it contributes useful information to the task.

## 4 Projection Model

From Table 2 we can see that the English Wiki-based taggers are better than the Bulgarian and Korean ones, which is due to the abundance and completeness of English data in Wikipedia. In such circumstances, previous research has shown that one can project annotations from English to the more resource-poor language (Yarowsky et al., 2001). Here we follow the approach of Feng et al. (2004) to train a log-linear model for projection.

Note that the Wiki-based taggers do not require training data and can be applied to any sentences from Wikipedia articles. The projection model described in this section and the Semi-CRF model described in Section 5 are trained using annotated data. They can be applied to tag foreign sentences in English-foreign sentence pairs extracted from Wikipedia.

The task of projection is re-cast as a ranking task, where for each source entity  $S_i$ , we rank all possible candidate target entity spans  $T_j$  and select the best span as corresponding to this source entity. Each target span is labeled with the NE label of the corresponding source entity. The probability distribution over target spans  $T_j$  for a given source entity  $S_i$  is defined as follows:

$$p(S_i|T_j) = \frac{\exp(\lambda f(S_i, T_j))}{\sum_{j'} \exp(\lambda f(S_i, T'_j))}$$

where  $\lambda$  is a parameter vector, and  $\mathbf{f}(S_i, T_j)$  is a fea-

ture vector for the candidate entity pair.

From this formulation we can see that a fixed set of English source entities  $S_i$  is required as input. The model projects these entities to corresponding foreign entities. We train and evaluate the projection model using 10-fold cross-validation on the dataset from Table 1. For training, we use the human-annotated gold English entities and the manually-specified entity alignments to derive corresponding target entities. At test time we use the local+global Wiki-based tagger to define the English entities and we don't use the manually annotated alignments.

### 4.1 Features

We present the features for this model in a lot of detail since analogous feature types are also used in our final direct semi-CRF model. The features are grouped into four categories.

#### Word alignment features

We exploit a feature set based on HMM word alignments in both directions (Och and Ney, 2000). To define the features we make use of the posterior alignment link probabilities as well as the most likely (Viterbi) alignments. The posterior probabilities are the probabilities of links in both directions given the source and target sentences:  $P(a_i = j|s, t)$  and  $P(a_j = i|s, t)$ .

If a source entity consists of positions  $i_1, \dots, i_m$  and a potential corresponding target entity consists of positions  $j_1, \dots, j_n$ , the word-alignment derived features are:

- Probability that each word from one of the entities is aligned to a word from the other entity, estimated as:

$\prod_{i \in i_1 \dots i_m} \sum_{j \in j_1 \dots j_n} P(a_i = j|s, t)$  We use an analogous estimate for the probability in the other direction.

- Sum of posterior probabilities of links from words inside one entity to words outside another entity  $\sum_{i \in i_1 \dots i_m} (1 - \sum_{j \in j_1 \dots j_n} P(a_i = j | s, t))$ . Probabilities from the other HMM direction are estimated analogously.
- Indicator feature for whether the source and target entity can be extracted as a phrase pair according to the combined Viterbi alignments (grow-diag-final) and the standard phrase extraction heuristic (Koehn et al., 2003).

### Phonetic similarity features

These features measure the similarity between a source and target entity based on pronunciation. We utilize a transliteration model (Cherry and Suzuki, 2009), trained from pairs of English person names and corresponding foreign language names, extracted from Wikipedia. The transliteration model can return an  $n$ -best list of transliterations of a foreign string, together with scores. For example the top 3 transliterations in English of the Bulgarian equivalent of “Igor Tudor” from Figure 1 are *Igor Twoodor*, *Igor Twoodore*, and *Igore Twoodore*.

We estimate phonetic similarity between a source and target entity by computing Levenshtein and other distance metrics between the source entity and the closest transliteration of the target (out of a 10-best list of transliterations). We use normalized and un-normalized Levenshtein distance. We also use a BLEU-type measure which estimates character  $n$ -gram overlap.

### Position/Length features

These report relative length and position of the English and foreign entity following (Feng et al., 2004).

### Wiki-based tagger features

These features look at the degree of match between the source and target entities based on the tags assigned to them by the local and global Wiki-taggers for English and the foreign language, and by the Stanford tagger for English. These are indicator features separate for the different source-target tagger combinations, looking at whether the taggers agree in their assignments to the candidate entities.

## 4.2 Model Evaluation

We evaluate the tagging F-measure for projection models on the English-Bulgarian and English-Korean datasets. 10-fold cross-validation was used to estimate model performance. The foreign language NE F-measure is reported in Table 3. The best Wiki-based tagger performance is shown on the last line as a baseline (repeated from Table 2).

We present a detailed evaluation of the model to gain understanding of the strengths and limitations of the projection approach and to motivate our direct semi-CRF model. To give an estimate of the upper bound on performance for the projection model, we first present two oracles. The goal of the oracles is to estimate the impact of two sources of error for the projection model: the first is the error in detecting English entities, and the second is the error in determining the corresponding foreign entity for a given English entity.

The first oracle ORACLE1 has access to the gold-standard English entities and gold-standard word alignments among English and foreign words. For each source entity, ORACLE1 selects the longest foreign language sequence of words that could be extracted in a phrase pair coupled with the source entity word sequence (according the standard phrase extraction heuristic (Koehn et al., 2003)), and labels it with the label of the source entity. Note that the word alignments do not uniquely identify the corresponding foreign phrase for each English phrase and some error is possible due to this. The performance of this oracle is closely related to the percentage of linked source-target entities reported in Table 1. The second oracle ORACLE2 provides the performance of the projection model when gold-standard source entities are known, but the corresponding target entities still have to be determined by the projection model (gold-standard alignments are not known). In other words, ORACLE2 is the projection model with all features, where in the test set we provide the gold standard English entities as input. The performance of ORACLE2 is determined by the error in automatic word alignment and in determining phonetic correspondence. As we can see the drop due to this error is very large, especially on Korean, where performance drops from 90.0 to 81.9 F-measure.

The next section in the Table presents the perfor-

Method	English-Bulgarian			English-Korean		
	Prec	Rec	F1	Prec	Rec	F1
ORACLE1	98.3	92.9	95.5	95.5	85.1	90.0
ORACLE2	96.7	86.3	91.2	90.5	74.7	81.9
PM-WF	71.7	80.0	75.7	85.1	72.2	78.1
PM+WF	73.6	81.3	77.2	87.6	74.9	<b>80.8</b>
Wiki-tagger	86.8	79.9	<b>83.2</b>	89.5	57.3	69.9

Table 3: English-Bulgarian and English-Korean Projection tagger performance.

mance of non-oracle projection models, which do not have access to any manually labeled information. The local+global Wiki-based tagger is used to define English entities, and only automatically derived alignment information is used. PM+WF is the projection model using all features. The line above, PM-WF represents the projection model without the Wiki-tagger derived features, and is included to show that the gain from using these features is substantial. The difference in accuracy between the projection model and ORACLE2 is very large, and is due to the error of the Wiki-based English taggers. The drop for Bulgarian is so large that the best projection model PM+WF does not reach the performance of 83.2 achieved by the baseline Wiki-based tagger. When source entities are assigned with error for this language pair, projecting entity annotations from the source is not better than using the target Wiki-based annotations directly. For Korean while the trend in model performance is similar as oracle information is removed, the projection model achieves substantially better performance (80.8 vs 69.9) due to the much larger difference in performance between the English and Korean Wiki-based taggers.

The drawback of the projection model is that it determines target entities only by assigning the best candidate for each source entity. It cannot create target entities that do not correspond to source entities, it is not able to take into account multiple conflicting source NE taggers as sources of information, and it does not make use of target sentence context and entity consistency constraints. To address these shortcomings we propose a direct semi-CRF model, described in the next section.

## 5 Semi-CRF Model

Semi-Markov conditional random fields (semi-CRFs) are a generalization of CRFs. They assign labels to segments of an input sequence  $\mathbf{x}$ , rather than

to individual elements  $x_i$  and features can be defined on complete segments. We apply Semi-CRFs to learn a NE tagger for labeling foreign sentences in the context of corresponding source sentences with existing NE annotations.

The semi-CRF defines a distribution over foreign sentence labeled segmentations (where the segments are named entities with their labels, or segments of length one with label “NONE”). To formally define the distribution, we introduce some notation following Sarawagi and Cohen (2005):

Let  $\mathbf{s} = \langle s_1, \dots, s_p \rangle$  denote a segmentation of the foreign sentence  $\mathbf{x}$ , where a segment  $s_j = \langle t_j, u_j, y_j \rangle$  is determined by its start position  $t_j$ , end position  $u_j$ , and label  $y_j$ . Features are defined on segments and adjacent segment labels. In our application, we only use features on segments. The features on segments can also use information from the corresponding English sentence  $\mathbf{e}$  along with external annotations on the sentence pair  $\mathbf{A}$ .

The feature vector for each segment can be denoted by  $F(j, \mathbf{s}, \mathbf{x}, \mathbf{e}, \mathbf{A})$  and the weight vector for features by  $\mathbf{w}$ . The probability of a segmentation is then defined as:

$$P(\mathbf{s}|\mathbf{x}, \mathbf{e}, \mathbf{A}) = \frac{\sum_j \exp \mathbf{w}' F(j, \mathbf{s}, \mathbf{x}, \mathbf{e}, \mathbf{A})}{Z(\mathbf{x}, \mathbf{e}, \mathbf{A})}$$

In the equation above  $Z$  represents a normalizer summing over valid segmentations.

### 5.1 Features

We use both boolean and real-valued features in the semi-CRF model. Example features and their values are given in Table 4. The features are the ones that fire on the segment of length 1 containing the Bulgarian equivalent of the word “Split” and labeled with label GPE ( $t_j=13, u_j=13, y_j=GPE$ ), from the English-Bulgarian sentence pair in Figure 1.

The features look at the English and foreign sentence as well as external annotations **A**. Note that the semi-CRF model formulation does not require a fixed labeling of the English sentence. Different and possibly conflicting NE tags for candidate English and foreign sentence substrings according to the Wiki-based taggers and the Stanford tagger are specified as one type of external annotations (see Figure 2). Another annotation type is derived from HMM-based word alignments and the transliteration model described in Section 4. They provide two kinds of alignment links between English and foreign tokens: one based on the HMM-word alignments (posterior probability of the link in both directions), and another based on different character-based distance metrics between transliterations of foreign words and English words. The transliteration model and distance metrics were described in Section 4 as well. For the example Bulgarian correspondent of “Split” in the figure, the English “Split” is linked to it according to both the forward and backward HMM, and according to two out of the three transliteration distance measures. A third annotation type is automatically derived links between foreign candidate entity strings (sequences of tokens) and best corresponding English candidate entities. The candidate English entities are defined by the union of entities proposed by the Wiki-based taggers and the Stanford tagger. Note that these English candidate entities can be overlapping and inconsistent without harming the model. We link foreign candidate segments with English candidate entities based on the projection model described in Section 4 and trained on the same data. The projection model scores every source-target entity pair and selects the best source for each target candidate entity. For our example target segment, the corresponding source candidate entity is “Split”, labeled GPE by the local+global Wiki-tagger and by the global Wiki-tagger.

The features are grouped into three categories:

**Group 1. Foreign Wiki-based tagger features.** These features look at target segments and extract indicators of whether the label of the segment agrees with the label assigned by the local, global, and/or local+global wiki tagger. For the example segment from the sentence in Figure 1, since neither the local nor global tagger have assigned a label GPE, the first three features have value zero. In addition to tags on

the whole segment, we look at tag combinations for individual words within the segment as well as two words to the left and right outside the segment. In the first section in Table 4 we can see several feature types and their values for our example.

**Group 2. Foreign surface-based features.** These features look at orthographic properties of the words and distinguish several word types. The types are based on capitalization and also distinguish numbers and punctuation. In addition, we make use of word-clusters generated by JCluster.<sup>1</sup>

We look at properties of the individual words as well as the concatenation for all words in the segment. In addition, there are features for words two words to the left and two words to the right outside the segment. The second section in the Table shows several features of this type with their values.

**Group 3. Label match between English and aligned foreign entities.** These features look at the linked English segment for the candidate target segment and compare the tags assigned to the English segment by the different English taggers to the candidate target label. In addition to segment-level comparisons, they also look at tag assignments for individual source tokens linked to the individual target tokens (by word alignment and transliteration links). The last section in the Table contains sample features with their values. The feature SOURCE-E-WIKI-TAG-MATCH looks at whether the corresponding source entity has the same local+global Wiki-tagger assigned tag as the candidate target entity. The next two features look at the Stanford tagger and the global Wiki-tagger. The real-valued features like SCORE-SOURCE-E-WIKI-TAG-MATCH return the score of the matching between the source and target candidate entities (according to the projection model), if the labels match. In this way, more confident matchings can impact the target tags more than less confident ones.

## 5.2 Experimental results

Our main results are listed in Table 5. We perform 10-fold cross-validation as in the projection experiments. The best Wiki-based and projection models are listed as baselines at the bottom of the table.

<sup>1</sup>Software distributed by Joshua Goodman <http://research.microsoft.com/en-us/downloads/0183a49d-c86c-4d80-aa0d-53c97ba7350a/default.aspx>.



Method	English-Bulgarian			English-Korean		
	Prec	Rec	F1	Prec	Rec	F1
MONO	86.7	79.4	82.9	89.1	57.1	69.6
BI	90.1	83.3	86.6	88.6	79.8	84.0
MONO-ALL	94.7	86.2	90.3	90.2	84.3	87.2
BI-ALL-WT	95.7	87.6	91.5	92.4	87.6	89.9
BI-ALL	96.4	89.4	<b>92.8</b>	94.7	87.9	<b>91.2</b>
Wiki-tagger	86.8	79.9	<b>83.2</b>	89.5	57.3	69.9
PM+WF	73.6	81.3	77.2	87.6	74.9	<b>80.8</b>

Table 5: English-Bulgarian and English-Korean semi-CRF tagger performance.

Feature Description	Example Value
WIKI-TAG-MATCH	0
WIKI-GLOBAL-TAG-MATCH	0
WIKIGLOBAL-POSSIBLE-TAG	0
WIKI-TAG&LABEL	NONE&GPE
WIKI-GLOBAL-TAG&LABEL	NONE&GPE
FIRST-WORD-CAP	1
CONTAINS-NUMBER	0
PREV-WORD-CAP	0
WORD-TYPE&LABEL	Xxxx&GPE
WORD-CLUSTER& LABEL	101&GPE
SEGMENT-WORD-TYPE&LABEL	Xxxx&GPE
SEGMENT-WORD-CLUSTER&LABEL	Xxxx&GPE
SOURCE-E-WIKI-TAG-MATCH	1
SOURCE-E-STANFORD-TAG-MATCH	0
SOURCE-E-WIKI-GLOBAL-TAG-MATCH	1
SOURCE-E-POSSIBLE-GLOBAL	1
SOURCE-E-ALL-TAG-MATCH	0
SOURCE-W-FWA-TAG & LABEL	GPE & GPE
SOURCE-W-BWA-TAG & LABEL	GPE & GPE
SCORE-SOURCE-E-WIKI-TAG-MATCH	-0.009
SCORE-SOURCE-E-GLOBAL-TAG-MATCH	-0.009
SCORE-SOURCE-E-STANFORD-TAG-MATCH	-1

Table 4: Features with example values.

We look at performance using four sets of features: (i) Monolingual Wiki-tagger based, using only the features in Group 1 (MONO); (ii) Bilingual label match and Wiki-tagger based, using features in Groups 1 and 3 (BI); (iii) Monolingual all, using features in Groups 1 and 2 (MONO-ALL), and (iv) Bilingual all, using all features (BI-ALL). Additionally, we report performance of the full bilingual model with all features, but when English candidate entities are generated only according to the local+global Wiki-tagger (BI-ALL-WT).

The main results show that the full semi-CRF model greatly outperforms the baseline projection and Wiki-taggers. For Bulgarian, the F-measure of the full model is 92.8 compared to the best baseline result of 83.2. For Korean, the F-measure of the semi-CRF is 91.2, more than 10 points higher than the performance of the projection model.

Within the semi-CRF model, the contribution of English sentence context was substantial, leading to 2.5 point increase in F-measure for Bulgarian (92.8 versus 90.3 F-measure), and 4.0 point increase for Korean (91.2 versus 87.2).

The additional gain due to considering candidate source entities generated from all English taggers was 1.3 F-measure points for both language pairs (comparing models BI-ALL and BI-ALL-WT).

If we restrict the semi-CRF to use only features similar to the ones used by the projection model, we still obtain performance much better than that of the projection model: comparing BI to the projection model, we see gains of 9.4 points for Bulgarian, and 4 points for Korean. This is due to the fact that the semi-CRF is able to relax the assumption of one-to-one correspondence between source and target entities, and can effectively combine information from multiple source and target taggers.

We should note that the proposed method can only tag foreign sentences in English-foreign sentence pairs. The next step for this work is to train monolingual NE taggers for the foreign languages, which can work on text within or outside of Wikipedia. Preliminary results show performance of over 80 F-measure for such monolingual models.

## 6 Related Work

As discussed throughout the paper, our model builds upon prior work on Wikipedia metadata-based NE tagging (Richman and Schone, 2008) and cross-lingual projection for named entities (Feng et al., 2004). Other interesting work on aligning named entities in two languages is reported in (Huang and Vogel, 2002; Moore, 2003).

Our direct semi-CRF tagging approach is related to bilingual labeling models presented in previous

work (Burkett et al., 2010a; Smith and Smith, 2004; Snyder and Barzilay, 2008). All of these models jointly label aligned source and target sentences. In contrast, our model is not concerned with tagging English sentences but only tags foreign sentences in the context of English sentences. Compared to the joint log-linear model of Burkett et al. (2010a), our semi-CRF approach does not require enumeration of  $n$ -best candidates for the English sentence and is not limited to  $n$ -best candidates for the foreign sentence. It enables the use of multiple unweighted and overlapping entity annotations on the English sentence.

## 7 Conclusions

In this paper we showed that using resources from Wikipedia, it is possible to combine metadata-based approaches and projection-based approaches for inducing named entity annotations for foreign languages. We presented a direct semi-CRF tagging model for labeling foreign sentences in parallel sentence pairs, which outperformed projection by more than 10 F-measure points for Bulgarian and Korean.

## References

- David Burkett, John Blitzer, and Dan Klein. 2010a. Joint parsing and alignment with weakly synchronized grammars. In *Proceedings of NAACL*.
- David Burkett, Slav Petrov, John Blitzer, and Dan Klein. 2010b. Learning better monolingual models with unannotated bilingual text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 46–54, Uppsala, Sweden, July. Association for Computational Linguistics.
- Colin Cherry and Hisami Suzuki. 2009. Discriminative substring decoding for transliteration. In *EMNLP*, pages 1066–1075.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609, Portland, Oregon, USA, June. Association for Computational Linguistics.
- Donghui Feng, Yajuan Lv, and Ming Zhou. 2004. A new approach for English-Chinese named entity alignment. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing EMNLP*, pages 372–379.
- Jenny Finkel, Trond Grenager, and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*.
- Fei Huang and Stephan Vogel. 2002. Improved named entity translation and bilingual named entity extraction. In *ICMI*.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*, pages 127–133.
- Robert C. Moore. 2003. Learning translations of named-entity phrases from parallel corpora. In *EACL*.
- Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*.
- Alexander E. Richman and Patrick Schone. 2008. Mining wiki resources for multilingual named entity recognition. In *ACL*.
- Sunita Sarawagi and William W. Cohen. 2004. Semi-markov conditional random fields for information extraction. In *In Advances in Neural Information Processing Systems 17*, pages 1185–1192.
- Sunita Sarawagi and William W. Cohen. 2005. Semi-markov conditional random fields for information extraction. In *In Advances in Neural Information Processing Systems 17 (NIPS 2004)*.
- David A. Smith and Noah A. Smith. 2004. Bilingual parsing with factored estimation: using English to parse Korean. In *EMNLP*.
- Jason R. Smith, Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *HLT*, pages 403–411, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Benjamin Snyder and Regina Barzilay. 2008. Crosslingual propagation for morphological analysis. In *AAAI*.
- David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *HLT*.

# A Computational Approach to the Automation of Creative Naming

**Gözde Özbal**  
FBK-Irst / Trento, Italy  
gozbalde@gmail.com

**Carlo Strapparava**  
FBK-Irst / Trento, Italy  
strappa@fbk.eu

## Abstract

In this paper, we propose a computational approach to generate neologisms consisting of homophonic puns and metaphors based on the category of the service to be named and the properties to be underlined. We describe all the linguistic resources and natural language processing techniques that we have exploited for this task. Then, we analyze the performance of the system that we have developed. The empirical results show that our approach is generally effective and it constitutes a solid starting point for the automation of the naming process.

## 1 Introduction

A catchy, memorable and creative name is an important key to a successful business since the name provides the first image and defines the identity of the service to be promoted. A good name is able to state the area of competition and communicate the promise given to customers by evoking semantic associations. However, finding such a name is a challenging and time consuming activity, as only few words (in most cases only one or two) can be used to fulfill all these objectives at once. Besides, this task requires a good understanding of the service to be promoted, creativity and high linguistic skills to be able to play with words. Furthermore, since many new products and companies emerge every year, the naming style is continuously changing and creativity standards need to be adapted to rapidly changing requirements.

The creation of a name is both an art and a science (Keller, 2003). Naming has a precise methodology

and effective names do not come out of the blue. Although it might not be easy to perceive all the effort behind the naming process just based on the final output, both a training phase and a long process consisting of many iterations are certainly required for coming up with a good name.

From a practical point of view, naming agencies and branding firms, together with automatic name generators, can be considered as two alternative services that facilitate the naming process. However, while the first type is generally expensive and processing can take rather long, the current automatic generators are rather naïve in the sense that they are based on straightforward combinations of random words. Furthermore, they do not take semantic reasoning into account.

To overcome the shortcomings of these two alternative ways (i.e. naming agencies and naïve generators) that can be used for obtaining name suggestions, we propose a system which combines several linguistic resources and natural language processing (NLP) techniques to generate creative names, more specifically neologisms based on homophonic puns and metaphors. In this system, similarly to the previously mentioned generators, users are able to determine the category of the service to be promoted together with the features to be emphasized. Our improvement lies in the fact that instead of random generation, we take semantic, phonetic, lexical and morphological knowledge into consideration to automatize the naming process.

Although various resources provide distinct tips for inventing creative names, no attempt has been made to combine all means of creativity that can be used during the naming process. Furthermore, in addition to the devices stated by copywriters, there

might be other latent methods that these experts unconsciously use. Therefore, we consider the task of discovering and accumulating all crucial features of creativity to be essential before attempting to automatize the naming process. Accordingly, we create a gold standard of creative names and the corresponding creative devices that we collect from various sources. This resource is the starting point of our research in linguistic creativity for naming.

The rest of the paper is structured as follows. First, we review the state-of-the-art relevant to the naming task. Then, we give brief information about the annotation task that we have conducted. Later on, we describe the model that we have designed for the automatization of the naming process. Afterwards, we summarize the annotation task that we have carried out and analyze the performance of the system with concrete examples by discussing its virtues and limitations. Finally, we draw conclusions and outline ideas for possible future work.

## 2 Related Work

In this section, we will analyze the state of the art concerning the naming task from three different aspects: i) linguistic ii) computational iii) commercial.

### 2.1 Linguistic

Little research has been carried out to investigate the linguistic aspects of the naming mechanism. B. V. Bergh (1987) built a four-fold linguistic topology consisting of phonetic, orthographic, morphological and semantic categories to evaluate the frequency of linguistic devices in brand names. Bao et al. (2008) investigated the effects of relevance, connotation, and pronunciation of brand names on preferences of consumers. Klink (2000) based his research on the area of sound symbolism (i.e. “the direct linkage between sound and meaning” (Leanne Hinton, 2006)) by investigating whether the sound of a brand name conveys an inherent meaning and the findings showed that both vowels and consonants of brand names communicate information related to products when no marketing communications are available. Kohli et al. (2005) analyzed consumer evaluations of meaningful and non-meaningful brand names and the results suggested that non-meaningful brand names are evaluated less favorably than meaningful ones even after repeated exposure. Lastly, cog (2011) focused on the semantics of branding and based on the analysis of several

international brand names, it was shown that cognitive operations such as domain reduction/expansion, mitigation, and strengthening might be used unconsciously while creating a new brand name.

### 2.2 Computational

To the best of our knowledge, there is only one computational study in the literature that can be applied to the automatization of name generation. Stock and Strapparava (2006) introduce an acronym ironic re-analyzer and generator called HAHAcronym. This system both makes fun of existing acronyms, and produces funny acronyms that are constrained to be words of the given language by starting from concepts provided by users. HAHAcronym is mainly based on lexical substitution via semantic field opposition, rhyme, rhythm and semantic relations such as antonyms retrieved from WordNet (Stark and Riesefeld, 1998) for adjectives.

As more naïve solutions, automatic name generators can be used as a source of inspiration in the brainstorming phase to get ideas for good names. As an example, [www.business-name-generators.com](http://www.business-name-generators.com) randomly combines abbreviations, syllables and generic short words from different domains to obtain creative combinations. The domain generator on [www.namestation.com](http://www.namestation.com) randomly generates name ideas and available domains based on alliterations, compound words and custom word lists. Users can determine the prefix and suffix of the names to be generated. The brand name generator on [www.netsubstance.com](http://www.netsubstance.com) takes keywords as inputs and here users can configure the percentage of the shifting of keyword letters. Lastly, the mechanism of [www.naming.net](http://www.naming.net) is based on name combinations among common words, Greek and Latin prefixes, suffixes and roots, beginning and ending word parts and rhymes. A shortcoming of these kinds of automatic generators is that random generation can output so many bad suggestions and users have to be patient to find the name that they are looking for. In addition, these generations are based on straightforward combinations of words and they do not include a mechanism to also take semantics into account.

### 2.3 Commercial

Many naming agencies and branding firms<sup>1</sup> provide professional service to aid with the naming of new

<sup>1</sup>e.g. [www.eatmywords.com](http://www.eatmywords.com), [www.designbridge.com](http://www.designbridge.com), [www.ahundredmonkeys.com](http://www.ahundredmonkeys.com)

products, domains, companies and brands. Such services generally require customers to provide brief information about the business to be named, fill in questionnaires to learn about their markets, competitors, and expectations. In the end, they present a list of name candidates to be chosen from. Although the resulting names can be successful and satisfactory, these services are very expensive and the processing time is rather long.

### 3 Dataset and Annotation

In order to create a gold standard for linguistic creativity in naming, collect the common creativity devices used in the naming process and determine the suitable ones for automation, we conducted an annotation task on a dataset of 1000 brand and company names from various domains (Özbal et al., 2012). These names were compiled from a book dedicated to brand naming strategies (Botton and Cegarra, 1990) and various web resources related to creative naming such as [adslogans.co.uk](http://adslogans.co.uk) and [brandsandtags.com](http://brandsandtags.com).

Our list contains names which were invented via various creativity methods. While the creativity in some of these names is independent of the context and the names themselves are sufficient to realize the methods used (e.g. alliteration in *Peak Performance*, modification of one letter in *Vimeo*), for some of them the context information such as the description of the product or the area of the company is also necessary to fully understand the methods used. For instance, *Thanks a Latte* is a coffee bar name where the phonetic similarity between “lot” and “latte” (a coffee type meaning “milk” in Italian) is exploited. The name *Caterpillar*, which is an earth-moving equipment company, is used as a metaphor. Therefore, we need extra information regarding the domain description in addition to the names. Accordingly, while building our dataset, we conducted two separate branches of annotation. The first branch required the annotators to fill in the domain description of the names in question together with their etymologies if required, while the second asked them to determine the devices of creativity used in each name.

In order to obtain the list of creativity devices, we collected a total of 31 attributes used in the naming process from various resources including academic papers, naming agents, branding and advertisement experts. To facilitate the task for the annotators,

we subsumed the most similar attributes when required. Adopting the four-fold linguistic topology suggested by Bergh et al. (B. V. Bergh, 1987), we mapped these attributes into phonetic, orthographic, morphological and semantic categories. The phonetic category includes attributes such as rhyme (i.e. repetition of similar sounds in two or more words - e.g. *Etch-a-sketch*) and reduplication (i.e. repeating the root or stem of a word or part of it exactly or with a slight change - e.g. *Teenie Weenie*), while the orthographic category consists of devices such as acronyms (e.g. *BMW*) and palindromes (i.e. words, phrases, numbers that can be read the same way in either direction e.g. *Honda “Civic”*). The third category is the morphology which contains affixation (i.e. forming different words by adding morphemes at the beginning, middle or end of words - e.g. *Nutella*) and blending (i.e. forming a word by blending sounds from two or more distinct words and combining their meanings - e.g. *Wikipedia* by blending “Wiki” and “encyclopedia”). Finally, the semantic category includes attributes such as metaphors (i.e. Expressing an idea through the image of another object - e.g. *Virgin*) and punning (i.e. using a word in different senses or words with sound similarity to achieve specific effect such as humor - e.g. *Thai Me Up* for a Thai restaurant).

### 4 System Description

The resource that we have obtained after the annotation task provides us with a starting point to study and try to replicate the linguistic and cognitive processes behind the creation of a successful name. Accordingly, we have made a systematic attempt to replicate these processes, and implemented a system which combines methods and resources used in various areas of Natural Language Processing (NLP) to create neologisms based on homophonic puns and metaphors. While the variety of creativity devices is actually much bigger, our work can be considered as a starting point to investigate which kinds of technologies can successfully be exploited in which way to support the naming process. The task that we deal with requires: 1) reasoning of relations between entities and concepts; 2) understanding the desired properties of entities determined by users; 3) identifying semantically related terms which are also consistent with the objectives of the advertisement; 4) finding terms which are suitable metaphors for the properties that need to be emphasized; 5) reasoning

about phonetic properties of words; 6) combining all this information to create natural sounding neologisms.

In this section, we will describe in detail the work flow of the system that we have designed and implemented to fulfill these requirements.

#### 4.1 Specifying the category and properties

Our design allows users to determine the category of the product/brand/company to be advertised (e.g. shampoo, car, chocolate) optionally together with the properties (e.g. softening, comfortable, additive) that they want to emphasize. In the current implementation, categories are required to be nouns while properties are required to be adjectives. These inputs that are specified by users constitute the main *ingredients* of the naming process. After the determination of these ingredients, several techniques and resources are utilized to enlarge the ingredient list, and thereby to increase the variety of new and creative names.

#### 4.2 Adding common sense knowledge

After the word defining the category is determined by the user, we need to automatically retrieve more information about this word. For instance, if the category has been determined as “shampoo”, we need to learn that “it is used for washing hair” or “it can be found in the bathroom”, so that all this extra information can be included in the naming process. To achieve that, we use ConceptNet (Liu and Singh, 2004), which is a semantic network containing common sense, cultural and scientific knowledge. This resource consists of nodes representing concepts which are in the form of words or short phrases of natural language, and labeled relations between them.

ConceptNet has a closed class of relations expressing connections between concepts. After the analysis of these relations according to the requirements of the task, we have decided to use the ones listed in Table 1 together with their description in the second column. The third column states whether the category word should be the first or second argument of the relation in order for us to consider the new word that we discover with that relation. Since, for instance, the relations *MadeOf(milk, \*)* and *MadeOf(\*, milk)* can be used for different goals (the former to obtain the ingredients of milk, and the latter to obtain products containing milk), we

Relation	Description	#	POS
HasA	What does it possess?	1	n
PartOf	What is it part of?	2	n
UsedFor	What do you use it for?	1	n,v
AtLocation	Where would you find it?	2	n
MadeOf	What is it made of?	1	n
CreatedBy	How do you bring it into existence?	1	n
HasSubevent	What do you do to accomplish it?	2	v
Causes	What does it make happen?	1	n,v
Desires	What does it want?	1	n,v
CausesDesire	What does it make you want to do?	1	n,v
HasProperty	What properties does it have?	1	a
ReceivesAction	What can you do to it?	1	v

Table 1: ConceptNet relations.

need to make this differentiation. Via ConceptNet 5, the latest version of ConceptNet, we obtain a list of relations such as *AtLocation(shampoo, bathroom)*, *UsedFor(shampoo, clean)* and *MadeOf(shampoo, perfume)* with the query word “shampoo”. We add all the words appearing in relations with the category word to our ingredient list. Among these new words, multiwords are filtered out since most of them are noisy and for our task a high precision is more important than a high recall.

Since sense information is not provided, one of the major problems in utilizing ConceptNet is the difficulty in disambiguating the concepts. In our current design, we only consider the most common senses of words. As another problem, the part-of-speech (POS) information is not available in ConceptNet. To handle this problem, we have determined the required POS tags of the new words that can be obtained from the relations with an additional goal of filtering out the noise. These tags are stated in the fourth column of Table 1.

#### 4.3 Adding semantically related words

To further increase the size of the ingredient list, we utilize another resource called WordNet (Miller, 1995), which is a large lexical database for English. In WordNet, nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms called synsets. Each synset in WordNet expresses a different concept and they are connected to each other with lexical, semantic and conceptual relations.

We use the *direct hypernym* relation of WordNet to retrieve the superordinates of the category word (e.g. *cleansing agent*, *cleanser* and *cleaner* for the category word *shampoo*). We prefer to use this relation of WordNet instead of the relation “IsA” in

ConceptNet to avoid getting too general words. Although we can obtain only the direct hypernyms in WordNet, no such mechanism exists in ConceptNet. In addition, while WordNet has been built by linguists, ConceptNet is built from the contributions of many thousands of people across the Web and naturally it also contains a lot of noise.

In addition to the direct hypernyms of the category word, we increase the size of the ingredient list by adding synonyms of the category word, the new words coming from the relations and the properties determined by the user.

It should be noted that we do not consider any other statistical or knowledge based techniques for semantic relatedness. Although they would allow us to discover more concepts, it is difficult to understand if and how these concepts pertain to the context. In WordNet we can decide what relations to explore, with the result of a more precise process with possibly less recall.

#### 4.4 Retrieving metaphors

A metaphor is a figure of speech in which an implied comparison is made to indicate how two things that are not alike in most ways are similar in one important way. Metaphors are common devices for evocation, which has been found to be a very important technique used in naming according to the analysis of our dataset.

In order to generate metaphors, we start with the set of properties determined by the user and adopt a similar technique to the one proposed by (Veale, 2011). In this work, to metaphorically ascribe a property to a term, stereotypes for which the property is culturally salient are intersected with stereotypes to which the term is pragmatically comparable. The stereotypes for a property are found by querying on the web with the simile pattern “as  $\langle property \rangle$  as \*”. Unlike the proposed approach, we do not apply any intersection with comparable stereotypes since the naming task should favor further terms to the category word in order to exaggerate, to evoke and thereby to be more effective.

The first constituent of our approach uses the pattern “as  $\langle property \rangle$  as \*” with the addition of “ $\langle property \rangle$  like \*”, which is another important block for building similes. Given a property, these patterns are harnessed to make queries through the web api of Google Suggest. This service performs auto-completion of search queries based on popu-

lar searches. Although top 10 (or fewer) suggestions are provided for any query term by Google Suggest, we expand these sets by adding each letter of the alphabet at the end of the provided phrase. Thereby, we obtain 10 more suggestions for each of these queries. Among the metaphor candidates that we obtain, we filter out multiwords to avoid noise as much as possible. Afterwards, we conduct a lemmatization process on the rest of the candidates. From the list of lemmas, we only consider the ones which appear in WordNet as a noun. Although the list that we obtain in the end has many potentially valuable metaphors (e.g. *sun*, *diamond*, *star*, *neon* for the property *bright*), it also contains a lot of uncommon and unrelated words (e.g. *downlaod*, *myspace*, *house*). Therefore, we need a filtering mechanism to remove the noise and keep only the best metaphors.

To achieve that, the second constituent of the metaphor retrieval mechanism makes a query in ConceptNet with the given property. Then, all the nouns coming from the relations in the form of *HasProperty*(\* , *property*) are collected to find words having that property. The POS check to obtain only nouns is conducted with a look-up in WordNet as before. It should be noted that this technique would not be enough to retrieve metaphors alone since it can also return noise (e.g. *blouse*, *idea*, *color*, *homeschooler* for the property *bright*).

After we obtain two different lists of metaphor candidates with the two mechanisms mentioned above, we take the intersection of these lists and consider only the words appearing in both lists as metaphors. In this manner, we aim to remove the noise coming from each list and obtain more reliable metaphors. To illustrate, for the same example property *bright*, the metaphors obtained at the end of the process are *sun*, *light* and *day*.

#### 4.5 Generating neologisms

After the ingredient list is complete, the phonetic module analyzes all ingredient pairs to generate neologisms with possibly homophonic puns based on phonetic similarity.

To retrieve the pronunciation of the ingredients, we utilize the CMU Pronouncing Dictionary (Lenzo, 2007). This resource is a machine-readable pronunciation dictionary of English which is suitable for uses in speech technology, and it contains over 125,000 words together with their transcriptions. It has mappings from words to their pronunciations

Input		Successful output		Unsuccessful output	
Category	Properties	Word	Ingredients	Word	Ingredients
bar	irish lively wooden traditional warm hospitable friendly	beertender barty giness	bartender, beer party, bar guinness, gin	barkplace barl bark	workplace, bar girl, bar work, bar
perfume	attractive strong intoxicating unforgettable feminine mystic sexy audacious provocative	mysticious bussling mysteelious	mysterious, mystic buss, puzzling mysterious, steel	provocadeepe	provocative, deep
sunglasses	cool elite though authentic cheap sporty	spectacools electacles polarice	spectacles, cool spectacles, elect polarize, ice	spocleang	sporting, clean
restaurant	warm elegant friendly original italian tasty cozy modern	eatalian pastarant peatza	italian, eat restaurant, pasta pizza, eat	dusta hometess	pasta, dust hostess, home
shampoo	smooth bright soft volumizing hydrating quality	fragrinse cleansun	fragrance, rinse cleanser, sun	furl sasun	girl, fur satin, sun

Table 2: A selection of succesful and unsuccessful neologisms generated by the model.

and the current phoneme set contains 39 phonemes based on the ARPAbet symbol set, which has been developed for speech recognition uses. We conducted a mapping from the ARPAbet phonemes to the international phonetic alphabet (IPA) phonemes and we grouped the IPA phonemes based on the phoneme classification documented in IPA. More specifically, we grouped the ones which appear in the same category such as p-b, t-d and s-z for the consonants; i-y and e-ø for the vowels.

After having the pronunciation of each word in the ingredient list, shorter pronunciation strings are compared against the substrings of longer ones. Among the different possible distance metrics that can be applied for calculating the phonetic similarity between two pronunciation strings, we have chosen the Levenshtein distance (Levenshtein, 1966). This distance is a metric for measuring the amount of difference between two sequences, defined as the minimum number of edits required for the transformation of one sequence into the other. The allowable edit operations for this transformation are insertion, deletion, or substitution of a single character. For example, the Levenshtein distance between the strings “kitten” and “sitting” is 3, since the following three edits change one into the other, and there is no way to do it with fewer than three edits: kitten → sitten (substitution of ‘k’ with ‘s’), sitten → sittin (substitution of ‘e’ with ‘i’), sittin → sitting (insertion of ‘g’ at the end). For the distance calculation, we employ relaxation by giving a smaller penalty for the

phonemes appearing in the same phoneme groups mentioned previously. We normalize each distance by the length of the pronunciation string considered for the distance calculation and we only allow the combination of word pairs that have a normalized distance score less than 0.5, which was set empirically.

Since there is no one-to-one relationship between letters and phonemes and no information about which phoneme is related to which letter(s) is available, it is not straightforward to combine two words after determining the pairs via Levenshtein distance calculation. To solve this issue, we use the Berkeley word aligner<sup>2</sup> for the alignment of letters and phonemes. The Berkeley Word Aligner is a statistical machine translation tool that automatically aligns words in a sentence-aligned parallel corpus. To adapt this tool according to our needs, we split all the words in our dictionary into letters and their mapped pronunciation to their phonemes, so that the aligner could learn a mapping from phonemes to characters. The resulting alignment provides the information about from which index to which index the replacement of the substring of a word should occur. Accordingly, the substring of the word which has a high phonetic similarity with a specific word is replaced with that word. As an example, if the first ingredient is *bright* and the second ingredient is *light*, the name *blight* can be obtained at the end of

<sup>2</sup><http://code.google.com/p/berkeleyaligner/>



this process.

#### 4.6 Checking phonetic likelihood

To check the likelihood and well-formedness of the new string after the replacement, we learn a 3-gram language model with absolute smoothing. For learning the language model, we only consider the words in the CMU pronunciation dictionary which also exist in WordNet. This filtering is required in order to eliminate a large number of non-English trigrams which would otherwise cause too high probabilities to be assigned to very unlikely sequences of characters. We remove the words containing at least one trigram which is very unlikely according to the language model. The threshold to determine the unlikely words is set to the probability of the least frequent trigram observed in the training data.

### 5 Evaluation

We evaluated the performance of our system with a manual annotation in which 5 annotators judged a set of neologisms along 4 dimensions: 1) appropriateness, i.e. the number of ingredients (0, 1 or 2) used to generate the neologism which are appropriate for the input; 2) pleasantness, i.e. a binary decision concerning the conformance of the neologism to the sound patterns of English; 3) humor/wittiness, i.e. a binary decision concerning the wittiness of the neologism; 4) success, i.e. an assessment of the fitness of the neologism as a name for the target category/properties (unsuccessful, neutral, successful).

To create the dataset, we first compiled a list of 50 categories by selecting 50 hyponyms of the synset *consumer goods* in WordNet. To determine the properties to be underlined, we asked two annotators to state the properties that they would expect to have in a product or company belonging to each category in our category list. Then, we merged the answers coming from the two annotators to create the final set of properties for each category.

Although our system is actually able to produce a limitless number of results for a given input, we limited the number of outputs for each input to reduce the effort required for the annotation task. Therefore, we implemented a ranking mechanism which used a hybrid scoring method by giving equal weights to the language model and the normalized phonetic similarity. Among the ranked neologisms for each input, we only selected the top 20 to build the dataset. It should be noted that for some input

	Dimension			
	APP	PLE	HUM	SUX
2	9.54	0	0	27.04
3	33.3	25.34	32.77	49.52
4	41.68	38.6	34.57	18.77
5	15.48	36.06	32.66	4.67
3+	90.46	100	100	72.96

Table 3: Inter-annotator agreement (in terms of majority class, MC) on the four annotation dimensions.

combinations the system produced less than 20 neologisms. Accordingly, our dataset consists of a total number of 50 inputs and 943 neologisms.

To have a concrete idea about the agreement between annotators, we calculated the majority class for each dimension. With 5 annotators, a majority class greater than or equal to 3 means that the absolute majority of the annotators agreed on the same decision. Table 3 shows the distribution of majority classes along the four dimensions of the annotation. For pleasantness (PLE) and humor (HUM), the absolute majority of the annotators (i.e. 3/5) agreed on the same decision in 100% of the cases, while for appropriateness (APP) the figure is only slightly lower. Concerning success, arguably the most subjective of the four dimensions, in 27% of the cases it is not possible to take a majority decision. Nevertheless, in almost 73% of the cases the absolute majority of the annotators agreed on the annotation of this dimension.

Table 4 shows the micro and macro-average of the percentage of cases in which at least 3 annotators have labeled the ingredients as appropriate (APP), and the neologisms as pleasant (PLE), humorous (HUM) or successful (SUX). The system selects appropriate ingredients in approximately 60% of the cases, and outputs pleasant, English-sounding names in  $\sim 87\%$  of the cases. Almost one name out of four is labeled as successful by the majority of the annotators, which we regard as a very positive result considering the difficulty of the task. Even though we do not explicitly try to inject humor in the neologisms, more than 15% of the generated names turn out to be witty or amusing. The system managed to generate at least one successful name for all 50 input categories and at least one witty name for 42. As expected, we found out that there is a very high correlation (91.56%) between the appropriateness of the

Accuracy	Dimension			
	APP	PLE	HUM	SUX
micro	59.60	87.49	16.33	23.86
macro	60.76	87.01	15.86	24.18

Table 4: Accuracy of the generation process along the four dimensions.

ingredients and the success of the name. A successful name is also humorous in 42.67% of the cases, while 62.34% of the humorous names are labeled as successful. This finding confirms our intuition that amusing names have the potential to be very appealing to the customers. In more than 76% of the cases, a humorous name is the product of the combination of appropriate ingredients.

In Table 2, we show a selection of successful and unsuccessful outputs generated for the category and the set of properties listed under the block of columns labeled as *Input* according to the majority of annotators (i.e. 3 or more). As an example of positive outcomes, we can focus on the columns under *Successful output* for the input target word *restaurant*. The model correctly selects the ingredients *eat* (a restaurant is *UsedFor* eating), *pizza* and *pasta* (which are found *AtLocation* restaurant) to generate an appropriate name. The three “palatable” neologisms generated are *eatalian* (from the combination of *eat* and *Italian*), *pastarant* (*pasta* + *restaurant*) and *peatza* (*pizza* + *eat*). These three suggestions are amusing and have a nice ring to them. As a matter of fact, it turns out that the name *Eatalian* is actually used by at least one real Italian restaurant located in Los Angeles, CA<sup>3</sup>.

For the same set of stimuli, the model also selects some ingredients which are not really related to the use-case, e.g., *dust* and *hostess* (both of which can be found *AtLocation* restaurant) and *home* (a synonym for *plate*, which can be found *AtLocation* restaurant, in the baseball jargon). With these ingredients, the model produces the suggestion *dusta* which sounds nice but has a negative connotation, and *hometess* which can hardly be associated to the input category.

A rather common class of unsuccessful outputs include words that, by pure chance, happen to be already existing in English. In these cases, no actual neologism is generated. Sometimes, the generated

words have rather unpleasant or irrelevant meanings, as in the case of *bark* for *bar*. Luckily enough, these kinds of outputs can easily be eliminated by filtering out all the output words which can already be found in an English dictionary or which are found to have a negative valence with state-of-the-art techniques (e.g. SentiWordNet (Esuli and Sebastiani, 2006)). Another class of negative results includes neologisms generated from ingredients that the model cannot combine in a good English-sounding neologism (e.g. *spocleang* from *sporting* and *clean* for *sunglasses* or *sasun* from *satin* and *sun* for *shampoo*).

## 6 Conclusion

In this paper, we have focused on the task of automating the naming process and described a computational approach to generate neologisms with homophonic puns based on phonetic similarity. This study is our first step towards the systematic emulation of the various creative devices involved in the naming process by means of computational methods.

Due to the complexity of the problem, a unified model to handle all the creative devices at the same time seems outside the reach of the current state-of-the-art NLP techniques. Nevertheless, the resource that we collected, together with the initial implementation of this model should provide a good starting point for other researchers in the area. We believe that our contribution will motivate other research teams to invest more effort in trying to tackle the related research problems.

As future work, we plan to improve the quality of the output by considering word sense disambiguation techniques to reduce the effect of inappropriate ingredients. We also want to extend the model to include multiword ingredients and to generate not only words but also short phrases. Then, we would like to focus on other classes of creative devices, such as affixation or rhyming. Lastly, we plan to make the system that we have developed publicly available and collect user feedback for further development and improvement.

## Acknowledgments

The authors were partially supported by a Google Research Award.

<sup>3</sup><http://www.eataliancafe.com/>

## References

- L. Oliver B. V. Bergh, K. Adler. 1987. Linguistic distinction among top brand names. *Journal of Advertising Research*, pages 39–44.
- Yeqing Bao, Alan T Shao, and Drew Rivers. 2008. Creating new brand names: Effects of relevance, connotation, and pronunciation. *Journal of Advertising Research*, 48(1):148.
- Marcel Botton and Jean-Jack Cegarra, editors. 1990. *Le nom de marque*. Paris McGraw Hill.
2011. Cognitive tools for successful branding. *Applied Linguistics*, 32:369–388.
- Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. pages 417–422.
- Kevin Lane Keller. 2003. *Strategic brand management: building, measuring and managing brand equity*. New Jersey: Prentice Hall.
- Richard R. Klink. 2000. Creating brand names with meaning: The use of sound symbolism. *Marketing Letters*, 11(1):5–20.
- C Kohli, K Harich, and Lance Leuthesser. 2005. Creating brand identity: a study of evaluation of new brand names. *Journal of Business Research*, 58(11):1506–1515.
- John J. Ohala Leanne Hinton, Johanna Nichols. 2006. *Sound Symbolism*. Cambridge University Press.
- Kevin Lenzo. 2007. The cmu pronouncing dictionary. <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>.
- V. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710.
- H. Liu and P. Singh. 2004. Conceptnet — a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226.
- George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41.
- Gözde Özbal, Carlo Strapparava, and Marco Guerini. 2012. Brand Pitt: A corpus to explore the art of naming. In *Proceedings of the eighth international conference on Language Resources and Evaluation (LREC-2012)*, Istanbul, Turkey, May.
- Michael M. Stark and Richard F. Riesenfeld. 1998. Wordnet: An electronic lexical database. In *Proceedings of 11th Eurographics Workshop on Rendering*. MIT Press.
- Oliviero Stock and Carlo Strapparava. 2006. Laughing with HAHAcronym, a computational humor system. In *proceedings of the 21st national conference on Artificial intelligence - Volume 2*, pages 1675–1678. AAAI Press.
- Tony Veale. 2011. Creative language retrieval: A robust hybrid of information retrieval and linguistic creativity. In *Proceedings of ACL 2011*, Portland, Oregon, USA, June.

# Unsupervised Relation Discovery with Sense Disambiguation

Limin Yao    Sebastian Riedel    Andrew McCallum

Department of Computer Science

University of Massachusetts, Amherst

{lmyao, riedel, mccallum}@cs.umass.edu

## Abstract

To discover relation types from text, most methods cluster shallow or syntactic patterns of relation mentions, but consider only one possible sense per pattern. In practice this assumption is often violated. In this paper we overcome this issue by inducing clusters of pattern senses from feature representations of patterns. In particular, we employ a topic model to partition entity pairs associated with patterns into sense clusters using local and global features. We merge these sense clusters into semantic relations using hierarchical agglomerative clustering. We compare against several baselines: a generative latent-variable model, a clustering method that does not disambiguate between path senses, and our own approach but with only local features. Experimental results show our proposed approach discovers dramatically more accurate clusters than models without sense disambiguation, and that incorporating global features, such as the document theme, is crucial.

## 1 Introduction

Relation extraction (RE) is the task of determining semantic relations between entities mentioned in text. RE is an essential part of information extraction and is useful for question answering (Ravichandran and Hovy, 2002), textual entailment (Szpektor et al., 2004) and many other applications.

A common approach to RE is to assume that relations to be extracted are part of a predefined ontology. For example, the relations are given in knowledge bases such as Freebase (Bollacker et al., 2008) or DBpedia (Bizer et al., 2009). However, in many applications, ontologies do not yet exist or have low

coverage. Even when they do exist, their maintenance and extension are considered to be a substantial bottleneck. This has led to considerable interest in unsupervised relation discovery (Hasegawa et al., 2004; Banko and Etzioni, 2008; Lin and Pantel, 2001; Bollegala et al., 2010; Yao et al., 2011). Here, the relation extractor simultaneously discovers facts expressed in natural language, and the ontology into which they are assigned.

Many relation discovery methods rely exclusively on the notion of either shallow or syntactic patterns that appear between two named entities (Bollegala et al., 2010; Lin and Pantel, 2001). Such patterns could be sequences of lemmas and Part-of-Speech tags, or lexicalized dependency paths. Generally speaking, relation discovery attempts to cluster such patterns into sets of equivalent or similar meaning. Whether we use sequences or dependency paths, we will encounter the problem of polysemy. For example, a pattern such as “A beat B” can mean that person A wins over B in competing for a political position, as pair “(Hillary Rodham Clinton, Jonathan Tasini)” in “Sen Hillary Rodham Clinton beats rival Jonathan Tasini for Senate.” It can also indicate that an athlete A beat B in a sports match, as pair “(Dmitry Tursunov, Andy Roddick)” in “Dmitry Tursunov beat the best American player Andy Roddick.” Moreover, it can mean “physically beat” as pair “(Mr. Harris, Mr. Simon)” in “On Sept. 7, 1999, Mr. Harris fatally beat Mr. Simon.” This is known as *polysemy*. If we work with patterns alone, our extractor will not be able to differentiate between these cases.

Most previous approaches do not explicitly address this problem. Lin and Pantel (2001) assumes only one sense per path. In (Pantel et al., 2007), they augment each relation with its selectional pref-

erences, i.e. fine-grained entity types of two arguments, to handle polysemy. However, such fine-grained entity types come at a high cost. It is difficult to discover a high-quality set of fine-grained entity types due to unknown criteria for developing such a set. In particular, the optimal granularity of entity types depends on the particular pattern we consider. For example, a pattern like “A beat B” could refer to A winning a sports competition against B, or a political election. To differentiate between these senses we need types such as “Politician” or “Athlete”. However, for “A, the parent of B” we only need to distinguish between persons and organizations (for the case of the sub-organization relation). In addition, there are senses that just cannot be determined by entity types alone: Take the meaning of “A beat B” where A and B are both persons; this could mean A physically beats B, or it could mean that A defeated B in a competition.

In this paper we address the problem of polysemy, while we circumvent the problem of finding fine-grained entity types. Instead of mapping entities to fine-grained types, we directly induce pattern senses by clustering feature representations of pattern contexts, i.e. the entity pairs associated with a pattern. This allows us to employ not only local features such as words, but also global features such as the document and sentence themes.

To cluster the entity pairs of a single relation pattern into senses, we develop a simple extension to Latent Dirichlet Allocation (Blei et al., 2003). Once we have our pattern senses, we merge them into clusters of different patterns with a similar sense. We employ hierarchical agglomerative clustering with a similarity metric that considers features such as the entity arguments, and the document and sentence themes.

We perform experiments on New York Times articles and consider lexicalized dependency paths as patterns in our data. In the following we shall use the term path and pattern exchangeably. We compare our approach with several baseline systems, including a generative model approach, a clustering method that does not disambiguate between senses, and our approach with different features. We perform both automatic and manual evaluations. For automatic evaluation, we use relation instances in Freebase as ground truth, and employ two clustering

metrics, pairwise F-score and  $B^3$  (as used in coference). Experimental results show that our approach improves over the baselines, and that using global features achieves better performance than using entity type based features. For manual evaluation, we employ a set intrusion method (Chang et al., 2009). The results also show that our approach discovers relation clusters that human evaluators find coherent.

## 2 Our Approach

We induce pattern senses by clustering the entity pairs associated with a pattern, and discover semantic relations by clustering these sense clusters. We represent each pattern as a list of entity pairs and employ a topic model to partition them into different sense clusters using local and global features. We take each sense cluster of a pattern as an atomic cluster, and use hierarchical agglomerative clustering to organize them into semantic relations. Therefore, a semantic relation comprises a set of sense clusters of patterns. Note that one pattern can fall into different semantic relations when it has multiple senses.

### 2.1 Sense Disambiguation

In this section, we discuss the details of how we discover senses of a pattern. For each pattern, we form a clustering task by collecting all entity pairs the pattern connects. Our goal is to partition these entity pairs into sense clusters. We represent each pair by the following features.

**Entity names:** We use the surface string of the entity pair as features. For example, for pattern “A play B”, pairs which contain B argument “Mozart” could be in one sense, whereas pairs which have “Mets” could be in another sense.

**Words:** The words between and around the two entity arguments can disambiguate the sense of a path. For example, “A’s parent company B” is different from “A’s largest company B” although they share the same path “A’s company B”. The former describes the sub-organization relationship between two companies, while the latter describes B as the largest company in a location A. The two words to the left of the source argument, and to the right of the destination argument also help sense discovery. For example, in “Mazurkas played by Anna Kijanowska, pianist”, “pianist” tells us pattern “A played by B”

takes the “music” sense.

**Document theme:** Sometimes, the same pattern can express different relations in different documents, depending on the document’s theme. For instance, in a document about politics, “A defeated B” is perhaps about a politician that won an election against another politician. While in a document about sports, it could be a team that won against another team in a game, or an athlete that defeated another athlete. In our experiments, we use the meta-descriptors of a document as side information and train a standard LDA model to find the theme of a document. See Section 3.1 for details.

**Sentence theme:** A document may cover several themes. Moreover, sometimes the theme of a document is too general to disambiguate senses. We therefore also extract the theme of a sentence as a feature. Details are in 3.1.

We call entity name and word features local, and the two theme features global.

We employ a topic model to discover senses for each path. Each path  $p_i$  forms a document, and it contains a list of entity pairs co-occurring with the path in the tuples. Each entity pair is represented by a list of features  $f_k$  as we described. For each path, we draw a multinomial distribution  $\theta$  over topics/senses. For each feature of an entity pair, we draw a topic/sense from  $\theta_{p_i}$ . Formally, the generative process is as follows:

$$\begin{aligned} \theta_{p_i} &\sim \text{Dirichlet}(\alpha) \\ \phi_z &\sim \text{Dirichlet}(\beta) \\ z_e &\sim \text{Multinomial}(\theta_{p_i}) \\ f_k &\sim \text{Multinomial}(\phi_{z_e}) \end{aligned}$$

Assume we have  $m$  paths and  $l$  entity pairs for each path. We denote each entity pair of a path as  $e(p_i) = (f_1, \dots, f_n)$ . Hence we have:

$$\begin{aligned} P(e_1(p_i), e_2(p_i), \dots, e_l(p_i) | z_1, z_2, \dots, z_l) \\ = \prod_{j=1}^l \prod_{k=1}^n p(f_k | z_j) p(z_j) \end{aligned}$$

We assume the features are conditionally independent given the topic assignments. Each feature is generated from a multinomial distribution  $\phi$ . We use Dirichlet priors on  $\theta$  and  $\phi$ . Figure 1 shows the graphical representation of this model.

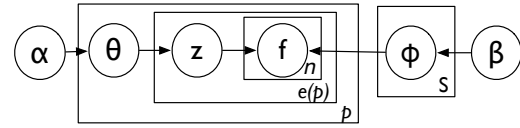


Figure 1: Sense-LDA model.

This model is a minor variation on standard LDA and the difference is that instead of drawing an observation from a hidden topic variable, we draw multiple observations from a hidden topic variable. Gibbs sampling is used for inference. After inference, each entity pair of a path is assigned to one topic. One topic is one sense. Entity pairs which share the same topic assignments form one sense cluster.

## 2.2 Hierarchical Agglomerative Clustering

After discovering sense clusters of paths, we employ hierarchical agglomerative clustering (HAC) to discover semantic relations from these sense clusters. We apply the complete linkage strategy and take cosine similarity as the distance function. The cutting threshold is set to 0.1.

We represent each sense cluster as one vector by summing up features from each entity pair in the cluster. The weight of a feature indicates how many entity pairs in the cluster have the feature. Some features may get larger weights and dominate the cosine similarity. We down-weight these features. For example, we use binary features for word “defeat” in sense clusters of pattern “A defeat B”. The two theme features are extracted from generative models, and each is a topic number.

Our approach produces sense clusters for each path and semantic relation clusters of the whole data. Table 1 and 2 show some example output.

## 3 Experiments

We carry out experiments on New York Times articles from years 2000 to 2007 (Sandhaus, 2008). Following (Yao et al., 2011), we filter out noisy documents and use natural language packages to annotate the documents, including NER tagging (Finkel et al., 2005) and dependency parsing (Nivre et al., 2004). We extract dependency paths for each pair of named entities in one sentence. We use their lemmas

Path	20:sports	30:entertainment	25:music/art
A play B	Americans, Ireland Yankees, Angels Ecuador, England Redskins, Detroit Red Bulls, F.C. Barcelona	Jean-Pierre Bacri, Jacques Rita Benton, Gay Head Dance Jeanie, Scrabble Meryl Streep, Leilah Kevin Kline, Douglas Fairbanks	Daniel Barenboim, recital of Mozart Mr. Rose, Ballade Gil Shaham, Violin Romance Ms. Golabek, Steinways Bruce Springsteen, Saints
<b>doc theme</b>	sports	music books television	music theater
<b>sen theme</b>	game yankees	theater production book film show	music reviews opera
<b>lexical words</b>	beat victory num-num won	played plays directed artistic	director conducted production
<b>entity names</b>	-	r:theater	r:theater r:hall r:york l:opera

Table 1: Example sense clusters produced by sense disambiguation. For each sense, we randomly sample 5 entity pairs. We also show top features for each sense. Each row shows one feature type, where “num” stands for digital numbers, and prefix “l:” for source argument, prefix “r:” for destination argument. Some features overlap with each other. We manually label each sense for easy understanding. We can see the last two senses are close to each other. For two theme features, we replace the theme number with the top words. For example, the document theme of the first sense is Topic30, and Topic30 has top words “sports”.

relation	paths
entertainment	A, who play B:30; A play B:30; star A as B:30
sports	lead A to victory over B:20; A play to B:20; A play B:20; A’s loss to B:20; A beat B:20; A trail B:20; A face B:26; A hold B:26; A play B:26; A acquire (X) from B:26; A send (X) to B:26;
politics	A nominate B:39; A name B:39; A select B:39; A name B:42; A select B:42; A ask B:42; A choose B:42; A nominate B:42; A turn to B:42;
law	A charge B:39; A file against B:39; A accuse B:39; A sue B:39

Table 2: Example semantic relation clusters produced by our approach. For each cluster, we list the top paths in it, and each is followed by “:number”, indicating its sense obtained from sense disambiguation. They are ranked by the number of entity pairs they take. The column on the left shows sense of each relation. They are added manually by looking at the sense numbers associated with each path.

for words on the dependency paths. Each entity pair and the dependency path which connects them form a tuple.

We filter out paths which occur fewer than 200 times and use some heuristic rules to filter out paths which are unlikely to represent a relation, for example, paths in which both arguments take the syntactic role “doj” (direct objective) in the dependency path. In such cases both arguments are often part of a coordination structure, and it is unlikely that they are related. In summary, we collect about one million tuples, 1300 patterns and half million named entities. In terms of named entities, the data is very sparse. On average one named entity occurs four times.

### 3.1 Feature Extraction

For the entity name features, we split each entity string of a tuple into tokens. Each token is a fea-

ture. The source argument tokens are augmented with prefix “l:”, and the destination argument tokens with prefix “r:”. We use tokens to encourage overlap between different entities.

For the word features, we extract all the words between the two arguments, removing stopwords and the words with capital letters. Words with capital letters are usually named entities, and they do not tend to indicate relations. We also extract neighboring words of source and destination arguments. The two words to the left of the source argument are added with prefix “lc:”. Similarly the two words to the right of the destination arguments are added with prefix “rc:”.

Each document in the NYT corpus is associated with many descriptors, indicating the topic of the document. For example, some documents are labeled as “Sports”, “Dallas Cowboys”, “New York Giants”, “Pro Football” and so on. Some are labeled

as “Politics and Government”, and “Elections”. We shall extract a theme feature for each document from these descriptors. To this end we interpret the descriptors as words in documents, and train a standard LDA model based on these documents. We pick the most frequent topic as the theme of a document.

We also train a standard LDA model to obtain the theme of a sentence. We use a bag-of-words representation for a document and ignore sentences from which we do not extract any tuples. The LDA model assigns each word to a topic. We count the occurrences of all topics in one sentence and pick the most frequent one as its theme. This feature captures the intuition that different words can indicate the same sense, for example, “film”, “show”, “series” and “television” are about “entertainment”, while “coach”, “game”, “jets”, “giants” and “season” are about “sports”.

### 3.2 Sense clusters and relation clusters

For the sense disambiguation model, we set the number of topics (senses) to 50. We experimented with other numbers, but this setting yielded the best results based on our automatic evaluation measures. Note that a path has a multinomial distribution over 50 senses but only a few senses have non-zero probabilities.

We look at some sense clusters of paths. For path “A play B”, we examine the top three senses, as shown in Table 1. The last two senses “entertainment” and “music” are close. Randomly sampling some entity pairs from each of them, we find that the two sense clusters are precise. Only 1% of pairs from the sense cluster “entertainment” should be assigned to the “music” sense. For the path “play A in B” we discover two senses which take the most probabilities: “sports” and “art”. Both clusters are precise. However, the “sports” sense may still be split into more fine-grained sense clusters. In “sports”, 67% pairs mean “play another team in a location” while 33% mean “play another team in a game”.

We also closely investigate some relation clusters, shown in Table 2. Both the first and second relation contain path “A play B” but with different senses. For the second relation, most paths state “play” relations between two teams, while a few of them express relations of teams acquiring players from

other teams. For example, the entity pair “(Atlanta Hawks, Dallas Mavericks)” mentioned in sentence “The Atlanta Hawks acquired point guard Anthony Johnson from the Dallas Mavericks.” This is due to that they share many entity pairs of team-team.

### 3.3 Baselines

We compare our approach against several baseline systems, including a generative model approach and variations of our own approach.

**Rel-LDA:** Generative models have been successfully applied to unsupervised relation extraction (Rink and Harabagiu, 2011; Yao et al., 2011). We compare against one such model: An extension to standard LDA that falls into the framework presented by Yao et al. (2011). Each document consists of a list of tuples. Each tuple is represented by features of the entity pair, as listed in 2.1, and the path. For each document, we draw a multinomial distribution over relations. For each tuple, we draw a relation topic and independently generate all the features. The intuition is that each document discusses one domain, and has a particular distribution over relations.

In our experiments, we test different numbers of relation topics. As the number goes up, precision increases whereas recall drops. We report results with 300 and 1000 relation topics.

**One sense per path (HAC):** This system uses only hierarchical clustering to discover relations, skipping sense disambiguation. This is similar to DIRT (Lin and Pantel, 2001). In DIRT, each path is represented by its entity arguments. DIRT calculates distributional similarities between different paths to find paths which bear the same semantic relation. It does not employ global topic model features extracted from documents and sentences.

**Local:** This system uses our approach (both sense clustering with topic models and hierarchical clustering), but without global features.

**Local+Type** This system adds entity type features to the previous system. This allows us to compare performance of using global features against entity type features. To determine entity types, we link named entities to Wikipedia pages using the Wikifier (Ratinov et al., 2011) package and extract categories from the Wikipedia page. Generally Wikipedia provides many types for one entity. For example, “Mozart” is



a *person*, *musician*, *pianist*, *composer*, and *catholic*. As we argued in Section 1, it is difficult to determine the right granularity of the entity types to use. In our experiments, we use all of them as features. In hierarchical clustering, for each sense cluster of a path, we pick the most frequent entity type as a feature. This approach can be seen as a proxy to ISP (Pantel et al., 2007), since selectional preferences are one way of distinguishing multiple senses of a path.

**Our Approach+Type** This system adds Wikipedia entity type features to our approach. The Wikipedia feature is the same as used in the previous system.

## 4 Evaluations

### 4.1 Automatic Evaluation against Freebase

We evaluate relation clusters discovered by all approaches against Freebase. Freebase comprises a large collection of entities and relations which come from varieties of data sources, including Wikipedia infoboxes. Many users also contribute to Freebase by annotating relation instances. We use coreference evaluation metrics: pairwise F-score and  $B^3$  (Bagga and Baldwin, 1998). Pairwise metrics measure how often two tuples which are clustered in one semantic relation are labeled with the same Freebase label. We evaluate approximately 10,000 tuples which occur in both our data and Freebase. Since our system predicts fine-grained clusters comparing against Freebase relations, the measure of recall is underestimated. The precision measure is more reliable and we employ F-0.5 measure, which places more emphasis on precision.

Matthews correlation coefficient (MCC) (Baldi et al., 2000) is another measure used in machine learning, which takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used when the classes are of very different sizes. In our case, the true negative number is 100 times larger than the true positive number. Therefore we also employ MCC, calculated as

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

The MCC score is between -1 and 1. The larger the better. In perfect predictions,  $FP$  and  $FN$  are 0, and the MCC score is 1. A random prediction results in score 0.

Table 3 shows the results of all systems. Our approach achieves the best performance in most measures. Without using sense disambiguation, the performance of hierarchical clustering decreases significantly, losing 17% in precision in the pairwise measure, and 15% in terms of  $B^3$ . The generative model approach with 300 topics achieves similar precision to the hierarchical clustering approach. With more topics, the precision increases, however, the recall of the generative model is much lower than those of other approaches. We also show the results of our approach without global document and sentence theme features (Local). In this case, both precision and recall decrease. We compare global features (Our approach) against Wikipedia entity type features (Local+Type). We see that using global features achieves better performance than using entity type based features. When we add entity type features to our approach, the performance does not increase. The entity type features do not help much is due to that we cannot determine which particular type to choose for an entity pair. Take pair “(Hillary Rodham Clinton, Jonathan Tasini)” as an example, choosing *politician* for both arguments instead of *person* will help.

We should note that these measures provide comparison between different systems although they are not accurate. One reason is the following: some relation instances should have multiple labels but they have only one label in Freebase. For example, instances of a relation that a person “was born in” a country could be labeled as “/people/person/place\_of\_birth” and as “/people/person/nationality”. This decreases the pairwise precision. Further discussion is in Section 4.3.

### 4.2 Path Intrusion

We also evaluate coherence of relation clusters produced by different approaches by creating path intrusion tasks (Chang et al., 2009). In each task, some paths from one cluster and an intruding path from another are shown, and the annotator’s job is to identify one single path which is out of place. For each path, we also show the annotators one example sentence. Three graduate students in natural language processing annotate intruding paths. For disagreements, we use majority voting. Table 4 shows one example intrusion task.

System	Pairwise				$B^3$		
	Prec.	Rec.	F-0.5	MCC	Prec.	Rec.	F-0.5
Rel-LDA/300	0.593	0.077	0.254	0.191	0.558	0.183	0.396
Rel-LDA/1000	0.638	0.061	0.220	0.177	0.626	0.160	0.396
HAC	0.567	0.152	0.367	0.261	0.523	<b>0.248</b>	0.428
Local	0.625	0.136	0.364	0.264	0.626	0.225	0.462
Local+Type	0.718	0.115	0.350	0.265	<b>0.704</b>	0.201	0.469
Our Approach	<b>0.736</b>	<b>0.156</b>	<b>0.422</b>	<b>0.314</b>	0.677	0.233	<b>0.490</b>
Our Approach+Type	0.682	0.110	0.334	0.250	0.687	0.199	0.460

Table 3: Pairwise and  $B^3$  evaluation for various systems. Since our systems predict more fine-grained clusters than Freebase, the recall measure is underestimated.

Path	Example sentence
A beat B	<b>Dmitry Tursunov</b> beat the best American player, <b>Andy Roddick</b>
A, who lose to B	<b>Sluman</b> , Loren Roberts (who lost a 1994 Open playoff to <b>Ernie Els</b> at Oakmont ...
A, who beat B	... offender seems to be the Russian <b>Mariya Sharapova</b> , who beat <b>Jelena Dokic</b>
<i>A, a broker at B</i>	<i><b>Robert Bewkes</b>, a broker at <b>UBS</b> for 12 years</i>
A meet B	<b>Howell</b> will meet <b>Geoff Ogilvy</b> , Harrington will face Davis Love III

Table 4: A path intrusion task. We show 5 paths and ask the annotator to identify one path which does not belong to the cluster. And we show one example sentence for each path. The entities (As and Bs) in the sentences are bold. And the italic row here indicates the intruder.

System	Correct
Rel-LDA/300	0.737
Rel-LDA/1000	0.821
HAC	0.852
Local+Type	0.773
Our approach	0.887

Table 5: Results of intruding tasks of all systems.

From Table 5, we see that our approach achieves the best performance. We concentrate on some intrusion tasks and compare the clusters produced by different systems.

The clusters produced by HAC (without sense disambiguation) is coherent if all the paths in one relation take a particular sense. For example, one task contains paths “A, director at B”, “A, specialist at B”, “A, researcher at B”, “A, B professor” and “A’s program B”. It is easy to identify “A’s program B” as an intruder when the annotators realize that the other four paths state the relation that people work in an educational institution. The generative model approach produces more coherent clusters when the number of relation topics increases.

The system which employs local and entity type features (Local+Type) produces clusters with low

coherence because the system puts high weight on types. For example, (*United States*, A talk with B, *Syria*) and (*Canada*, A defeat B, *United States*) are clustered into one relation since they share the argument types “country”-“country”. Our approach using the global theme features can correct such errors.

### 4.3 Error Analysis

We also closely analyze the pairwise errors that we encounter when comparing against Freebase labels. Some errors arise because one instance can have multiple labels, as we explained in Section 4.1. One example is the following: Our approach predicts that (*News Corporation*, buy, *MySpace*) and (*Dow Jones & Company*, the parent of, *The Wall Street Journal*) are in one relation. In Freebase, one is labeled as “/organization/parent/child”, the other is labeled as “/book/newspaper\_owner/newspapers\_owned”. The latter is a sub-relation of the former. We can overcome this issue by introducing hierarchies in relation labels.

Some errors are caused by selecting the incorrect sense for an entity pair of a path. For instance, we put (*Kenny Smith*, who grew up in, *Queens*) and (*Phil Jackson*, return to, *Los Angeles Lakers*) into

the “/people/person/place\_of\_birth” relation cluster since we do not detect the “sports” sense for the entity pair “(Phil Jackson, Los Angeles Lakers)”.

## 5 Related Work

There has been considerable interest in unsupervised relation discovery, including clustering approach, generative models and many other approaches.

Our work is closely related to DIRT (Lin and Pantel, 2001). Both DIRT and our approach represent dependency paths using their arguments. Both use distributional similarity to find patterns representing similar semantic relations. Based on DIRT, Pantel et al. (2007) addresses the issue of multiple senses per path by automatically learning admissible argument types where two paths are similar. They cluster arguments to fine-grained entity types and rank the associations of a relation with these entity types to discover selectional preferences. Selectional preferences discovery (Ritter et al., 2010; Seaghdha, 2010) can help path sense disambiguation, however, we show that using global features performs better than entity type features.

Our approach is also related to feature partitioning in cross-cutting model of lexical semantics (Reisinger and Mooney, 2011). And our sense disambiguation model is inspired by this work. There they partition features of words into views and cluster words inside each view. In our case, each sense of a path can be seen as one view. However, we allow different views to be merged since some views overlap with each other.

Hasegawa et al. (2004) cluster pairs of named entities according to the similarity of context words intervening between them. Hachey (2009) uses topic models to perform dimensionality reduction on features when clustering entity pairs into relations. Bollegala et al. (2010) employ co-clustering to find clusters of entity pairs and patterns jointly. All the approaches above neither deal with polysemy nor incorporate global features, such as sentence and document themes.

Open information extraction aims to discover relations independent of specific domains (Banko et al., 2007; Banko and Etzioni, 2008). They employ a self-learner to extract relation instances, but no attempt is made to cluster instances into relations.

Yates and Etzioni (2009) present RESOLVER for discovering relational synonyms as a post processing step. Our approach falls into the same category. Moreover, we explore path senses and global features for relation discovery.

Many generative probabilistic models have been applied to relation extraction. For example, varieties of topic models are employed for both open domain (Yao et al., 2011) and in-domain relation discovery (Chen et al., 2011; Rink and Harabagiu, 2011). Our approach employs generative models for path sense disambiguation, which achieves better performance than directly applying generative models to unsupervised relation discovery.

## 6 Conclusion

We explore senses of paths to discover semantic relations. We employ a topic model to partition entity pairs of a path into different sense clusters and use hierarchical agglomerative clustering to merge senses into semantic relations. Experimental results show our approach discovers precise relation clusters and outperforms a generative model approach and a clustering method which does not address sense disambiguation. We also show that using global features improves the performance of unsupervised relation discovery over using entity type based features.

## Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and the University of Massachusetts gratefully acknowledges the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, AFRL, or the US government.

## References

- Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *The First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*.

- Pierre Baldi, Søren Brunak, Yves Chauvin, Claus A. F. Andersen, and Henrik Nielsen. 2000. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics*, 16:412–424.
- Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Proceedings of ACL-08: HLT*.
- Michele Banko, Michael J Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of IJCAI2007*.
- Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009. DBpedia - a crystallization point for the web of data. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, pages 154–165.
- David Blei, Andrew Ng, and Michael Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, January.
- Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, New York, NY, USA. ACM.
- Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2010. Relational duality: Unsupervised extraction of semantic relations between entities on the web. In *Proceedings of WWW*.
- Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Proceedings of NIPS*.
- Harr Chen, Edward Benson, Tahira Naseem, and Regina Barzilay. 2011. In-domain relation discovery with meta-constraints via posterior regularization. In *Proceedings of ACL*.
- Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)*, pages 363–370, June.
- Benjamin Hachey. 2009. *Towards Generic Relation Extraction*. Ph.D. thesis, University of Edinburgh.
- Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *ACL*.
- Dekang Lin and Patrick Pantel. 2001. DIRT - Discovery of Inference Rules from Text. In *Proceedings of KDD*.
- J. Nivre, J. Hall, and J. Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of CoNLL*, pages 49–56.
- Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. 2007. ISP: Learning Inferential Selectional Preferences. In *Proceedings of NAACL HLT*.
- Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of ACL*.
- Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL*.
- Joseph Reisinger and Raymond J. Mooney. 2011. Cross-cutting models of lexical semantics. In *Proceedings of EMNLP*.
- Bryan Rink and Sanda Harabagiu. 2011. A generative model for unsupervised discovery of relations and argument classes from clinical texts. In *Proceedings of EMNLP*.
- Alan Ritter, Mausam, and Oren Etzioni. 2010. A Latent Dirichlet Allocation method for Selectional Preferences. In *Proceedings of ACL10*.
- Evan Sandhaus, 2008. *The New York Times Annotated Corpus*. Linguistic Data Consortium, Philadelphia.
- Diarmuid O Seaghdha. 2010. Latent variable models of selectional preference. In *Proceedings of ACL 10*.
- Idan Szepktor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP*.
- Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of EMNLP*.
- Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *Journal of Artificial Intelligence Research*, 34:255–296.

# Reducing Wrong Labels in Distant Supervision for Relation Extraction

Shingo Takamatsu

System Technologies Laboratories  
Sony Corporation

5-1-12 Kitashinagawa, Shinagawa-ku, Tokyo

Shingo.Takamatsu@jp.sony.com

Issei Sato and Hiroshi Nakagawa

Information Technology Center  
The University of Tokyo

7-3-1 Hongo, Bunkyo-ku, Tokyo

{sato@r., n3@}dl.itc.u-tokyo.ac.jp

## Abstract

In relation extraction, distant supervision seeks to extract relations between entities from text by using a knowledge base, such as Freebase, as a source of supervision. When a sentence and a knowledge base refer to the same entity pair, this approach heuristically labels the sentence with the corresponding relation in the knowledge base. However, this heuristic can fail with the result that some sentences are labeled wrongly. This noisy labeled data causes poor extraction performance. In this paper, we propose a method to reduce the number of wrong labels. We present a novel generative model that directly models the heuristic labeling process of distant supervision. The model predicts whether assigned labels are correct or wrong via its hidden variables. Our experimental results show that this model detected wrong labels with higher performance than baseline methods. In the experiment, we also found that our wrong label reduction boosted the performance of relation extraction.

## 1 Introduction

Machine learning approaches have been developed to address relation extraction, which is the task of extracting semantic relations between entities expressed in text. Supervised approaches are limited in scalability because labeled data is expensive to produce. A particularly attractive approach, called distant supervision (DS), creates labeled data by heuristically aligning entities in text with those in a knowledge base, such as Freebase (Mintz et al., 2009).

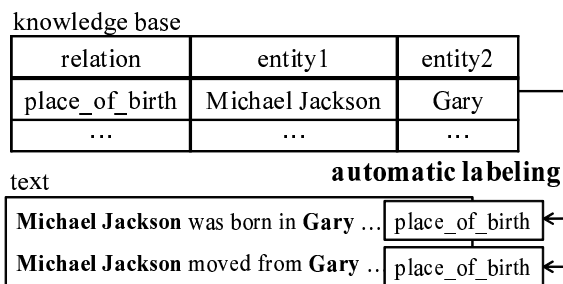


Figure 1: Automatic labeling by distant supervision. Upper sentence: correct labeling; lower sentence: incorrect labeling.

With DS it is assumed that if a sentence contains an entity pair in a knowledge base, such a sentence actually expresses the corresponding relation in the knowledge base.

However, the DS assumption can fail, which results in noisy labeled data and this causes poor extraction performance. An entity pair in a target text generally expresses more than one relation while a knowledge base stores a subset of the relations. The assumption ignores this possibility. For instance, consider the *place\_of\_birth* relation between *Michael Jackson* and *Gary* in Figure 1. The upper sentence indeed expresses the *place\_of\_birth* relation between the two entities. In DS *place\_of\_birth* is assigned to the sentence, and it becomes a useful training example. On the other hand, the lower sentence does not express this relation between the two entities, but the DS heuristic wrongly labels the sentence as expressing it.

Riedel et al. (2010) relax the DS assumption as at least one sentence containing an entity pair ex-

pressing the corresponding relation in the knowledge base. They cast the relaxed assumption as multi-instance learning. However, even the relaxed assumption can fail. The relaxation is equivalent to the DS assumption when a labeled pair of entities is mentioned once in a target corpus (Riedel et al., 2010). In fact, 91.7% of entity pairs appear only once in Wikipedia articles (see Section 7).

In this paper, we propose a method to reduce the number of wrong labels generated by DS without using either of these assumptions. Given the labeled corpus created with the DS assumption, we first predict whether each *pattern*, which frequently appears in text to express a relation (see Section 4), expresses a target relation. Patterns that are predicted not to express the relation are used to form a negative pattern list for removing wrong labels of the relation.

The main contributions of this paper are as follows:

- To make the pattern prediction, we propose a generative model that directly models the process of automatic labeling in DS. Without any strong assumptions like Riedel et al. (2010)'s, the model predicts whether each pattern expresses each relation via hidden variables (see Section 5).
- Our variational inference for our generative model lets us automatically calibrate parameters for each relation, which are sensitive to the performance (see Section 6).
- We applied our method to Wikipedia articles using Freebase as a knowledge base and found that (i) our model identified patterns expressing a given relation more accurately than baseline methods and (ii) our method led to better extraction performance than the original DS (Mintz et al., 2009) and MultiR (Hoffmann et al., 2011), which is a state-of-the-art multi-instance learning system for relation extraction (see Section 7).

## 2 Related Work

The increasingly popular approach, called distant supervision (DS), or weak supervision, utilizes a knowledge base to heuristically label a corpus (Wu and Weld, 2007; Bellare and McCallum, 2007; Pal

et al., 2007). Our work was inspired by Mintz et al. (2009) who used Freebase as a knowledge base by making the DS assumption and trained relation extractors on Wikipedia. Previous works (Hoffmann et al., 2010; Yao et al., 2010) have pointed out that the DS assumption generates noisy labeled data, but did not directly address the problem. Wang et al. (2011) applied a rule-based method to the problem by using popular entity types and keywords for each relation. In (Bellare and McCallum, 2007; Riedel et al., 2010; Hoffmann et al., 2011), they used multi-instance learning, which deals with uncertainty of labels, to relax the DS assumption. However, the relaxed assumption can fail when a labeled entity pair is mentioned only once in a corpus (Riedel et al., 2010). Our approach relies on neither of these assumptions.

Bootstrapping for relation extraction (Riloff and Jones, 1999; Pantel and Pennacchiotti, 2006; Carlson et al., 2010) is related to our method. In bootstrapping, seed entity pairs of the target relation are given in order to select reliable patterns, which are used to extract new entity pairs. To avoid the selection of unreliable patterns, bootstrapping introduces scoring functions for each pattern candidate. This can be applied to our approach, which seeks to reduce the number of unreliable patterns by using a set of given entity pairs. However, the bootstrapping-like approach suffers from sensitive parameters that are critical to its performance. Ideally, the parameters such as a threshold for scoring function should be determined for each relation, but there are no principled methods (Komachi et al., 2008). In our approach, parameters are calibrated for each relation by maximizing the likelihood of our generative model.

## 3 Knowledge-based Distant Supervision

In this section, we describe DS for relation extraction. We use the term *relation* as the relation between two entities. A *relation instance* is a tuple consisting of two entities and relation  $r$ . For example, *place\_of\_birth(Michael Jackson, Gary)* in Figure 1 is a relation instance.

Relation extraction seeks to extract relation instances from text. An entity is mentioned as a named entity in text. We extract a relation instance from a

single sentence. For example, from the upper sentence in Figure 1 we extract *place\_of\_birth*(*Michael Jackson, Gary*). Since two entities mentioned in a sentence do not always have a relation, we select entity pairs from a corpus when: (i) the path of the dependency parse tree between the corresponding two named entities in the sentence is no longer than 4 and (ii) the path does not contain a sentence-like boundary, such as a relative clause<sup>1</sup> (Banko et al., 2007; Banko and Etzioni, 2008). Banko and Etzioni (2008) found that a set of eight lexico-syntactic forms covers nearly 95% of relation phrases in their corpus. (Fader et al. (2011) found that this set covers 69% of their corpus). Our rule is designed to cover at least the eight lexico-syntactic forms. We use the entity pairs extracted by this rule.

DS uses a knowledge base to create labeled data for relation extraction by heuristically matching entity pairs. A *knowledge base* is a set of relation instances about predefined relations. For each sentence in the corpus, we extract all of its entity pairs. Then, for each entity pair, we try to retrieve the relation instances about the entity pair from the knowledge base. If we found such a relation instance, then the set of its relation, the entity pair, and the sentence is stored as a positive example. If not, then the set of the entity pair and the sentence is stored as a negative example. Features of an entity pair are extracted from the sentence containing the entity pair.

As mentioned in Section 1, the assumption of DS can fail, resulting in wrong assignments of a relation to sentences that do not express the relation. We call such assignments *wrong labels*. An example of a wrong label is *place\_of\_birth* assigned to the lower sentence in Figure 1.

## 4 Wrong Label Reduction

We define a *pattern* as the entity types of an entity pair<sup>2</sup> as well as the sequence of words on the path of the dependency parse tree from the first entity to the second one. For example, from “Michael Jackson was born in Gary” in Figure 1, the pattern “[Person] born in [Location]” is extracted. We use entity

<sup>1</sup>We reject sentence-like dependencies such as *ccomp*, *complm* and *mark*

<sup>2</sup>If we use a standard named entity tagger, the entity types are Person, Location, and Organization.

---

## Algorithm 1 Wrong Label Reduction

---

```

labeled data generated by DS:  $LD$ 
negative patterns for relation  $r$ :  $NegPat(r)$ 
for each entry  $(r, Pair, Sentence)$  in  $LD$  do
  pattern  $Pat \leftarrow$  the pattern from  $(Pair, Sentence)$ 
  if  $Pat \in NegPat(r)$  then
    remove  $(r, Pair, Sentence)$  from  $LD$ 
  end if
end for
return  $LD$ 

```

---

types to distinguish the sentences that express different relations with the same dependency path, such as “ABBA was formed in Stockholm.” and “ABBA was formed in 1970.”

Our aim is to remove wrong labels assigned to frequent patterns, which cause poor precision. Indeed, in our Wikipedia corpus, more than 6% of the sentences containing the pattern “[Person] moved to [Location]”, which does not express *place\_of\_death*, are labeled as *place\_of\_death*, and the labels assigned to these sentences hurt extraction performance (see Section 7.3.3). We would like to remove *place\_of\_death* from the sentences that contain this pattern.

In our method, we reduce the number of wrong labels as follows: (i) given a labeled corpus with the DS assumption, we first predict whether a pattern expresses a relation and then (ii) remove wrong labels using the negative pattern list, which is defined as patterns that are predicted not to express the relation. In the first step, we introduce the novel generative model that directly models DS’s labeling process and make the prediction (see Section 5). The second step is formally described in Algorithm 1. For relation extraction, we train a classifier for entity pairs using the resultant labeled data.

## 5 Generative Model

We now describe our generative model, which predicts whether a pattern expresses relation  $r$  or not via hidden variables. In this section, we consider relation  $r$  since parameters are conditionally independent if relation  $r$  and the hyperparameter are given.

An observation of our model is whether entity pair  $i$  appearing with pattern  $s$  in the corpus is labeled with relation  $r$  or not. Our binary observations are written as  $\mathbf{X}_r = \{(x_{rsi}) | s = 1, \dots, S, i =$

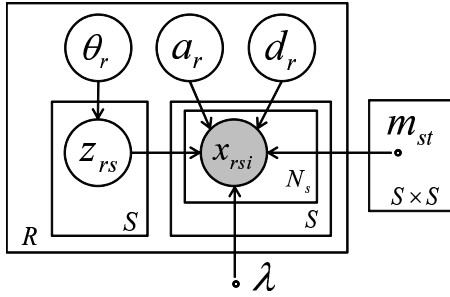


Figure 2: Graphical model representation of our model.  $R$  indicates the number of relations.  $S$  is the number of patterns.  $N_s$  is the number of entity pairs that appear with pattern  $s$  in the corpus.  $x_{rsi}$  is the observed variables. The circled variables except  $x_{rsi}$  are parameters or hidden variables.  $\lambda$  is the hyperparameter and  $m_{st}$  is constant. The boxes are “plates” representing replicates.

$1, \dots, N_s\}$ ,<sup>3</sup> where we define  $S$  to be the number of patterns and  $N_s$  to be the number of entity pairs appearing with pattern  $s$ . Note that we count an entity pair for given pattern  $s$  once even if the entity pair is mentioned with pattern  $s$  more than once in the corpus, because DS assigns the same relation to all mentions of the entity pair.

Given relation  $r$ , our model assumes the following generative process:

1. For each pattern  $s$ 
  - Choose whether  $s$  expresses relation  $r$  or not
$$z_{rs} \sim Be(\theta_r)$$
2. For each entity pair  $i$  appearing with pattern  $s$ 
  - Choose whether  $i$  is labeled or not
$$x_{rsi} \sim P(x_{rsi} | \mathbf{Z}_r, a_r, d_r, \lambda, \mathbf{M}),$$

where  $Be(\theta_r)$  is a Bernoulli distribution with parameter  $\theta_r$ ,  $z_{rs}$  is a binary hidden variable that is 1 if pattern  $s$  expresses relation  $r$  and 0 otherwise, and  $\mathbf{Z}_r = \{(z_{rs}) | s = 1, \dots, S\}$ . Given a value of  $z_{rs}$ , we model two kinds of probabilities: one for patterns that actually express relation  $r$ , i.e.,  $P(x_{rsi} = 1 | z_{rs} = 1)$ , and one for patterns that do not express  $r$ , i.e.,  $P(x_{rsi} = 1 | z_{rs} = 0)$ . The former is simply parameterized as  $0 \leq a_r \leq 1$ . We express the latter as  $b_{rs} = P(x_{rsi} = 1 | \mathbf{Z}_r, a_r, d_r, \lambda, \mathbf{M})$ , which is a function of  $\mathbf{Z}_r, a_r, d_r, \lambda$  and  $\mathbf{M}$ ; we explain its modeling in the following two subsections.

<sup>3</sup>Since a set of entity pairs appearing with pattern  $s$  is different,  $i$  should be written as  $i_s$ . For simplicity, however, we use  $i$  for each pattern.

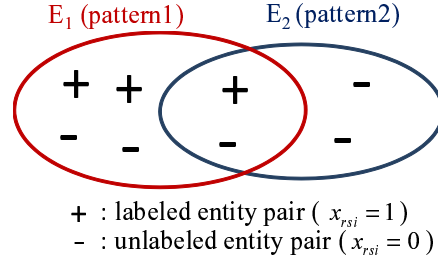


Figure 3: Venn diagram-like description.  $E_1$  and  $E_2$  are sets of entity pairs.  $E_1/E_2$  has 6/4 entity pairs because the 6/4 entity pairs appear with pattern 1/2 in the target corpus. Pattern 1 expresses relation  $r$  and pattern 2 does not. Elements in  $E_1$  are labeled with probability  $a_r = 3/6 = 0.5$ . Those in  $E_2$  are labeled with probability  $b_{r2} = a_r(|E_1 \cap E_2|/|E_2|) = 0.5(2/4) = 0.25$ .

The graphical model of our model is shown in Figure 2.

### 5.1 Example of Wrong Labeling

Using a simple example, we describe how we model  $b_{rs}$ , the probability with which DS assigns relation  $r$  to pattern  $s$  via entity pairs when pattern  $s$  does not express relation  $r$ .

Consider two patterns: pattern 1 that expresses relation  $r$  and pattern 2 that does not (i.e.,  $z_{r1} = 1$  and  $z_{r2} = 0$ ). We also assume that there are entity pairs that appear with pattern 1 as well as with pattern 2 in different places in the corpus (for example, *Michael Jackson* and *Gary* in Figure 1). When such entity pairs are labeled, relation  $r$  is assigned to pattern 1 and at the same time to wrong pattern 2. Such entity pairs are observed as elements in the intersection of the two sets of entity pairs,  $E_1$  and  $E_2$ . Here,  $E_s$  is the set of entity pairs that appear with pattern  $s$  in the corpus. This situation is described in Figure 3.

We model probability  $b_{r2}$  as follows. In  $E_1$ , an entity pair is labeled with probability  $a_r$ . We assume that entity pairs in the intersection,  $E_1 \cap E_2$ , are also labeled with  $a_r$ . From the viewpoint of  $E_2$ , entity pairs in its subset,  $E_1 \cap E_2$ , are labeled with  $a_r$ . Therefore,  $b_{r2}$  is modeled as

$$b_{r2} = a_r \frac{|E_1 \cap E_2|}{|E_2|},$$

where  $|E|$  denotes the number of elements in set  $E$ . An example of this calculation is shown in Figure 3.



We generalize the example in the next subsection.

## 5.2 Modeling of Probability $b_{rs}$

We model  $b_{rs}$  so that it is proportional to the number of entity pairs that are shared with correct patterns whose  $z_{rs} = 1$ , i.e.,

$$b_{rs} = a_r \frac{\left| \left( \bigcap_{\{t|z_{rt}=1, t \neq s\}} E_t \right) \cap E_s \right|}{|E_s|}, \quad (1)$$

where  $\cap$  indicates set intersections. However, the enumeration in Eq.1 requires  $O(SN_s^2)$  computational cost and a huge amount of memory to store all of the entity pairs. We approximate the right-hand side of Eq.1 as

$$b_{rs} \approx a_r \left( 1 - \prod_{t=1, t \neq s}^S \left( 1 - \frac{|E_t \cap E_s|}{|E_s|} \right)^{z_{rt}} \right).$$

This approximation is made, given the sizes of all  $E_s$ s and those of all intersections of two  $E_s$ s. This has a lower computational cost of  $O(S)$  and let us use less memory. We define  $S \times S$  matrix  $\mathbf{M}$  whose elements are  $m_{st} = |E_t \cap E_s|/|E_s|$ .

In reality, factors other than the process described in the previous subsection can cause wrong labeling (for example, errors in the knowledge base). We introduce a parameter  $0 \leq d_r \leq 1$  that covers such factors. Finally, we define  $b_{rs}$  as

$$b_{rs} \equiv a_r \left( \lambda \left( 1 - \prod_{t=1, t \neq s}^S (1 - m_{st})^{z_{rt}} \right) + (1 - \lambda) d_r \right), \quad (2)$$

where  $0 \leq \lambda \leq 1$  is the hyperparameter that controls how strongly  $b_{rs}$  is affected by the main labeling process explained in the previous subsection.

## 5.3 Likelihood

Given observation  $\mathbf{X}_r$ , the likelihood of our model is

$$\begin{aligned} P(\mathbf{X}_r | \theta_r, a_r, d_r, \lambda, \mathbf{M}) \\ = \sum_{\mathbf{Z}_r} P(\mathbf{Z}_r | \theta_r) P(\mathbf{X}_r | \mathbf{Z}_r, a_r, d_r, \lambda, \mathbf{M}), \end{aligned}$$

where

$$P(\mathbf{Z}_r | \theta_r) = \prod_{s=1}^S \theta_r^{z_{rs}} (1 - \theta_r)^{1 - z_{rs}}.$$

For each pattern  $s$ , we define  $n_{rs}$  as the number of entity pairs to which relation  $r$  is assigned (i.e.,  $n_{rs} = \sum_i x_{rsi}$ ).

$$\begin{aligned} p(\mathbf{X}_r | \mathbf{Z}_r, a_r, d_r, \lambda, \mathbf{M}) = \\ \prod_{s=1}^S \left\{ a_r^{n_{rs}} (1 - a_r)^{N_s - n_{rs}} \right\}^{z_{rs}} \\ \left\{ b_{rs}^{n_{rs}} (1 - b_{rs})^{N_s - n_{rs}} \right\}^{1 - z_{rs}}, \quad (3) \end{aligned}$$

where  $b_{rs}$  is in Eq.2.

## 6 Learning

We learn parameters  $a_r$ ,  $\theta_r$ , and  $d_r$  and infer hidden variables  $\mathbf{Z}_r$  by maximizing the log likelihood given  $\mathbf{X}_r$ . Estimated  $\mathbf{Z}_r$  is used to predict which patterns express relation  $r$ .

To infer  $z_{rs}$ , we would like to calculate the posterior probability of  $z_{rs}$ . However, this calculation is intractable because each  $z_{rs}$  depends on the others,  $\{(z_{rt}) | t \neq s\}$ , as shown in Eqs.2 and 3. This prevents us from using the EM algorithm. Instead, we apply variational approximation to the posterior distribution by using the following trial distribution:

$$Q(\mathbf{Z}_r | \Phi_r) = \prod_{s=1}^S \phi_{rs}^{z_{rs}} (1 - \phi_{rs})^{1 - z_{rs}},$$

where  $0 \leq \phi_{rs} \leq 1$  is a parameter for the trial distribution.

The following function  $F_r$  is a lower bound of the log likelihood, and maximizing this function with respect to  $\Phi_r$  is equivalent to minimizing the KL divergence between the trial distribution and the posterior distribution of  $\mathbf{Z}_r$ .

$$\begin{aligned} F_r = E_Q[\log P(\mathbf{Z}_r, \mathbf{X}_r | \theta_r, a_r, d_r, \lambda, \mathbf{M})] \\ - E_Q[\log Q(\mathbf{Z}_r | \Phi_r)]. \quad (4) \end{aligned}$$

$E_Q[\bullet]$  represents the expectation over trial distribution  $Q$ . We maximize function  $F_r$  with respect to the parameters instead of the log likelihood.

However, we need further approximation for two terms on expanding Eq.4. Both of the terms are expressed as  $E_Q[\log(f(\mathbf{Z}_r))]$ , where  $f(\mathbf{Z}_r)$  is a function of  $\mathbf{Z}_r$ . We apply the following approximation (Asuncion et al., 2009).

$$E_Q[\log(f(\mathbf{Z}_r))] \approx \log(E_Q[f(\mathbf{Z}_r)]).$$

This is based on the Taylor series of  $\log$  at  $E_Q[f(\mathbf{Z}_r)]$ . In our problem, since the second derivative is sufficiently small, we use the zeroth-order approximation.<sup>4</sup>

Our learning algorithm is derived by calculating the stationary condition of the resultant evaluation function with respect to each parameter. We have the exact solution for  $\theta_r$ . For each  $\phi_{rs}$  and  $d_r$ , we derive a fixed point iteration. We update  $a_r$  by using the steepest ascent. We update each parameter in turn while keeping the other parameters fixed. Parameter updating proceeds until a termination condition is met.

After learning, we have  $\phi_{rs}$  for each pair of relation  $r$  and pattern  $s$ . The greater the value of  $\phi_{rs}$  is, the more likely it is that pattern  $s$  expresses relation  $r$ . We set a threshold and determine  $z_{rs} = 0$  when  $\phi_{rs}$  is less than the threshold.

## 7 Experiments

We performed two sets of experiments.

**Experiment 1** aimed to evaluate the performance of our generative model itself, which predicts whether a pattern expresses a relation, given a labeled corpus created with the DS assumption.

**Experiment 2** aimed to evaluate how much our wrong label reduction in Section 4 improved the performance of relation extraction. In our method, we trained a classifier with a labeled corpus cleaned by Algorithm 1 using the negative pattern list predicted by the generative model.

### 7.1 Dataset

Following Mintz et al. (2009), we carried out our experiments using Wikipedia as the target corpus and Freebase (September, 2009, (Google, 2009)) as the knowledge base. We used more than 1,300,000 Wikipedia articles in the wex dump data (September, 2009, (Metaweb Technologies, 2009)). The properties of our data are shown in Table 1.

In Wikipedia articles, named entities were identified by anchor text linking to another article and starting with a capital letter (Yan et al., 2009). We applied Open NLP POS tagger<sup>5</sup> and MaltParser (Nivre et al., 2007) to sentences containing more

Table 1: Properties of Wikipedia dataset

documents	1,303,000
entity pairs	2,017,000
(matched to Freebase)	129,000
(with entity types)	913,000
frequent patterns	3,084
relations	24

than one named entity. We then extracted sentences containing related entity pairs with the method explained in Section 3. To match entity pairs, we used ID mapping between the dump data and Freebase. We used the most frequent 24 relations.

### 7.2 Experiment 1: Pattern Prediction

We compared our model with baseline methods in terms of ability to predict patterns that express a given relation.

The input of this task was  $\mathbf{X}_{r,s}$ , which expresses whether or not each entity pair appearing with each pattern is labeled with relation  $r$ , as explained in Section 5. In Experiment 1, since we needed entity types for patterns, we restricted ourselves to entities matched with Freebase, which also provides entity types for entities. We used patterns that appear more than 20 times in the corpus.

#### 7.2.1 Evaluation

We split the data into training data and test data. The training data was  $\mathbf{X}_{r,s}$  for 12 relations and the test data was that for the remaining 12 relations. The training data was used to calibrate parameters (see the following subsection for details). The test data was used for evaluation. We randomly split the data five times and took the average of the following evaluation values.

We evaluated the performance by precision, recall, and F value. They were calculated using gold standard data, which was constructed by hand. We manually selected patterns that actually express a target relation as positive patterns for the relation.<sup>6</sup> We averaged the evaluation values in terms of macro average over relations before averaging over the data splits.

<sup>6</sup>Patterns that ambiguously express the relation, for instance “[Person] in [Location]” for *place\_of\_birth*, were not selected as positive patterns.

<sup>4</sup>The first-order information becomes zero in this case.

<sup>5</sup><http://opennlp.sourceforge.net/>

Table 2: Averages of precision, recall, and F value in Experiment 1. The averages of threshold of RS(rank) and RS(value) were  $6.2 \pm 3.2$  and  $0.10 \pm 0.06$ , respectively. The averages of hyperparameters of PROP were  $0.84 \pm 0.05$  for  $\lambda$  and  $0.85 \pm 0.10$  for the threshold.

	Precision	Recall	F value
Baseline	0.339	1.000	0.458
RS(rank)	0.749	0.549	0.467
RS(value)	0.601	0.647	0.545
PROP	0.782	0.688	0.667

## 7.2.2 Methods

We compared the following methods:

**Baseline:** This method assigns relation  $r$  to a pattern when the pattern is mentioned with at least one entity pair corresponding to relation  $r$  in Freebase. This method is based on the DS assumption.

**Ratio-based Selection (RS):** Given relation  $r$  and pattern  $s$ , this method calculates  $n_{rs}/N_s$ , which is the ratio of the number of labeled entity pairs appearing with pattern  $s$  to the number of entity pairs including unlabeled ones. RS then selects the top  $n$  patterns (RS(rank)). We also tested a version using a real-valued threshold (RS(value)). In training, we selected the threshold that maximized the F value. Some bootstrapping approaches (Carlson et al., 2010) use a rank-based threshold like RS(rank).

**Proposed Model (PROP):** Using the training data, we determined the two hyperparameters,  $\lambda$  and the threshold to round  $\phi_{rs}$  to 1 or 0, so that they maximized the F value. When  $\phi_{rs}$  is greater than the threshold, we select pattern  $s$  as one expressing relation  $r$ .

## 7.2.3 Result and Discussion

The results of Experiment 1 are shown in Table 2. Our model achieved the best precision, recall, and F value. RS(value) had the second best F value, but it completely removed more than one infrequent relation on average in test sets. This is problematic for real situations. RS(rank) achieved the second highest precision. However, its recall, which is also important in our task, was the lowest and its F value was almost the same as naive Baseline.

The thresholds of RS, which directly affect their performance, should be calibrated for each relation, but it is hard to do this in advance. On the other

Table 3: Example of estimated  $\phi_{rs}$  for  $r = \textit{place\_of\_birth}$ . Entity types are omitted in patterns.  $n_{rs}/N_s$  is the ratio of the number of labeled entity pairs to the number of entity pairs appearing with pattern  $s$ .

pattern $s$	$n_{rs}/N_s$	$\phi_{rs}$	expresses $r$ ?
born in	0.512	0.999	true
actor from	0.480	0.999	true
elected Mayor of	0.384	0.855	false
family moved from	0.344	0.055	false
native of	0.327	0.999	true
grew in	0.162	0.000	false

hand, our model learns parameters such as  $a_r$  for each relation and thus the hyperparameter of our model does not directly affect its performance. This results in a high prediction performance.

Examples of estimated  $\phi_{rs}$ , the probability with which pattern  $s$  expresses relation  $r$ , are shown in Table 3. The pattern, “[Person] family moved from [Location]”, which does not express *place\_of\_birth*, had low  $\phi_{rs}$  in spite of having higher  $n_{rs}/N_s$  than the valid pattern “[Person] native of [Location]”. The former pattern had higher  $b_{rs}$ , the probability with which relation  $r$  is wrongly assigned to pattern  $s$  via entity pairs, because there were more entity pairs that appeared not only with this pattern but also with patterns that was predicted to express *place\_of\_birth*.

## 7.3 Experiment 2: Relation Extraction

We investigated the performance of relation extraction using our wrong label reduction, which uses the results of the pattern prediction.

Following Mintz et al. (2009), we performed an automatic held-out evaluation and a manual evaluation. In both cases, we used 400,000 articles for testing and the remaining 903,000 for training.

### 7.3.1 Configuration of Classifiers

Following Mintz et al. (2009), we used a multi-class logistic classifier optimized using L-BFGS with Gaussian regularization to classify entity pairs to the predefined 24 relations and NONE. In order to train the NONE class, we randomly picked 100,000 examples that did not match to Freebase as pairs. (Several entities in the examples matched and had entity types of Freebase.) In this experiment, we

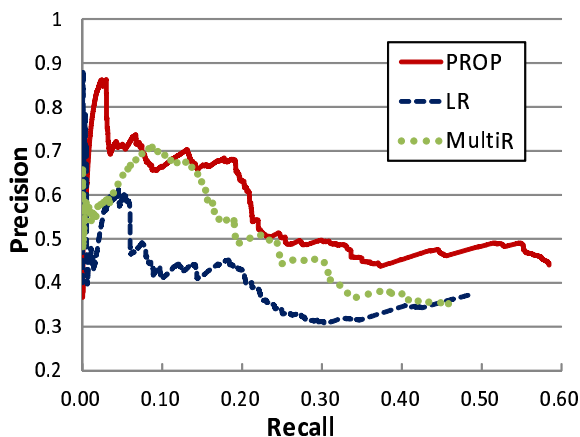


Figure 4: Precision-recall curves in held-out evaluation. Precision is reported at recall levels from 5 to 50,000.

used not only entity pairs matched to Freebase but also ones not matched to Freebase (i.e., entity pairs that do not have entity types). We used syntactic features (i.e., features obtained from the dependency parse tree of a sentence) and lexical features, and entity types, which essentially correspond to the ones developed by Mintz et al. (2009).

We compared the following methods: logistic regression with the labeled data cleaned by the proposed method (PROP), logistic regression with the standard DS labeled data (LR), and MultiR proposed in (Hoffmann et al., 2011) as a state-of-the-art multi-instance learning system.<sup>7</sup> For logistic regression, when more than one relation is assigned to a sentence, we simply copied the feature vector and created a training example for each relation. In PROP, we used training articles for pattern prediction.<sup>8</sup>

### 7.3.2 Held-out Evaluation

In the held-out evaluation, relation instances discovered from testing articles were automatically compared with those in Freebase. This let us calculate the precision of each method for the best  $n$  relation instances. The precisions are underestimated because this evaluation suffers from false negatives due to the incompleteness of Freebase. We changed  $n$  from 5 to 50,000 and measured precision and recall. Precision-recall curves for the held-out data are

<sup>7</sup>For MultiR, we used the authors’ implementation from <http://www.cs.washington.edu/homes/raphaelh/mr/>

<sup>8</sup>In Experiment 2 we set  $\lambda = 0.85$  and the threshold at 0.95.

Table 4: Averages of precisions at 50 for the most frequent 15 relations as well as example relations.

	PROP	MultiR	LR
<i>place_of_birth</i>	1.0	1.0	0.56
<i>place_of_death</i>	1.0	0.7	0.84
average	$0.89 \pm 0.14$	$0.83 \pm 0.21$	$0.82 \pm 0.23$

shown in Figure 4.

PROP achieved comparable or higher precision at most recall levels compared with LR and MultiR. Its performance at  $n = 50,000$  is much higher than that of the others. While our generative model does not use unlabeled examples as negative ones in detecting wrong labels, classifier-based approaches including MultiR do, suffering from false negatives.

### 7.3.3 Manual Evaluation

For manual evaluation, we picked the top ranked 50 relation instances for the most frequent 15 relations. The manually evaluated precisions averaged over the 15 relations are shown in table 4.

PROP achieved the best average precision. For *place\_of\_birth*, LR wrongly extracted entity pairs with “[Person] played with club [Location]”, which does not express the relation. PROP and MultiR avoided this mistake. For *place\_of\_death*, LR and MultiR wrongly extracted entity pairs with “[Person] moved to [Location]”. Multi-instance learning does not work for wrong labels assigned to entity pairs that appear only once in a corpus. In fact, 72% of entity pairs that appeared with this pattern and were wrongly labeled as *place\_of\_death* appeared only once in the corpus. Only PROP avoided mistakes of this kind because our method works in such situations.

## 8 Conclusion

We proposed a method that reduces the number of wrong labels created with the DS assumption, which is widely applied. Our generative model directly models the labeling process of DS and predicts patterns that are wrongly labeled with a relation. The predicted patterns are used for wrong label reduction. The experimental results show that this method successfully reduced the number of wrong labels and boosted the performance of relation extraction.

## References

- Arthur Asuncion, Max Welling, Padhraic Smyth, and Yee W. Teh. 2009. On smoothing and inference for topic models. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI '09)*, pages 27–34.
- Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '08)*, pages 28–36.
- Michele Banko, Michael J Cafarella, Stephen Soderl, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI '07)*, pages 2670–2676.
- Kedar Bellare and Andrew McCallum. 2007. Learning Extractors from Unlabeled Text using Relevant Databases. In *Sixth International Workshop on Information Integration on the Web (IIWeb '07)*.
- Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM '10)*, pages 101–110.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1535–1545.
- Google. 2009. Freebase data dumps. <http://download.freebase.com/datadumps/>.
- Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL '10)*, pages 286–295.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT '11)*, pages 541–550.
- Mamoru Komachi, Taku Kudo, Masashi Shimbo, and Yuji Matsumoto. 2008. Graph-based analysis of semantic drift in Espresso-like bootstrapping algorithms. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pages 1011–1020.
- Metaweb Technologies. 2009. Freebase wikipedia extraction (wex). <http://download.freebase.com/wex/>.
- Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP '09)*, pages 1003–1011.
- Joakim Nivre, Johan Hall, and Jens Nilsson. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 37:95–135.
- Chris Pal, Gideon Mann, and Richard Minerich. 2007. Putting semantic information extraction on the map: Noisy label models for fact extraction. In *Sixth International Workshop on Information Integration on the Web (IIWeb '07)*.
- Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL '06)*, pages 113–120.
- Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD '10)*, pages 148–163.
- Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479.
- Chang Wang, James Fan, Aditya Kalyanpur, and David Gondek. 2011. Relation extraction with relation topics. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1426–1436.
- Fei Wu and Daniel S. Weld. 2007. Autonomously semantifying wikipedia. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management (CIKM '07)*, pages 41–50.
- Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. 2009. Unsupervised relation extraction by mining wikipedia texts using information from the web. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP '09)*, pages 1021–1029.
- Limin Yao, Sebastian Riedel, and Andrew McCallum. 2010. Collective cross-document relation extraction without labelled data. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*, pages 1013–1023.

# Finding Salient Dates for Building Thematic Timelines

**Rémy Kessler**  
LIMSI-CNRS  
Orsay, France  
kessler@limsi.fr

**Xavier Tannier**  
Univ. Paris-Sud,  
LIMSI-CNRS  
Orsay, France  
xtannier@limsi.fr

**Caroline Hagège**  
Xerox Research Center Europe  
Meylan, France  
hagege@xrce.xerox.com

**Véronique Moriceau**  
Univ. Paris-Sud, LIMSI-CNRS  
Orsay, France  
moriceau@limsi.fr

**André Bittar**  
Xerox Research Center Europe  
Meylan, France  
bittar@xrce.xerox.com

## Abstract

We present an approach for detecting salient (important) dates in texts in order to automatically build event timelines from a search query (e.g. the name of an event or person, etc.). This work was carried out on a corpus of newswire texts in English provided by the Agence France Presse (AFP). In order to extract salient dates that warrant inclusion in an event timeline, we first recognize and normalize temporal expressions in texts and then use a machine-learning approach to extract salient dates that relate to a particular topic. We focused only on extracting the dates and not the events to which they are related.

## 1 Introduction

Our aim here was to build thematic timelines for a general domain topic defined by a user query. This task, which involves the extraction of important events, is related to the tasks of *Retrospective Event Detection* (Yang et al., 1998), or *New Event Detection*, as defined for example in Topic Detection and Tracking (TDT) campaigns (Allan, 2002).

The majority of systems designed to tackle this task make use of textual information in a bag-of-words manner. They use little temporal information, generally only using document metadata, such as the document creation time (DCT). The few systems that do make use of temporal information (such as the now discontinued Google timeline), only extract absolute, full dates (that feature a day, month and year). In our corpus, described in Section 3.1, we found that only 7% of extracted temporal expressions are absolute dates.

We distinguish our work from that of previous researchers in that we have focused primarily on extracted temporal information as opposed to other textual content. We show that using linguistic temporal processing helps extract important events in texts. Our system extracts a maximum of temporal information and uses only this information to detect salient dates for the construction of event timelines. Other types of content are used for initial thematic document retrieval. Output is a list of dates, ranked from most important to least important with respect to the given topic. Each date is presented with a set of relevant sentences.

We can see this work as a new, easily evaluable task of “date extraction”, which is an important component of timeline summarization.

In what follows, we first review some of the related work in Section 2. Section 3 presents the resources used and gives an overview of the system. The system used for temporal analysis is described in Section 4, and the strategy used for indexing and finding salient dates, as well as the results obtained, are given in Section 5<sup>1</sup>.

## 2 Related Work

The ISO-TimeML language (Pustejovsky et al., 2010) is a specification language for manual annotation of temporal information in texts, but, to the best of our knowledge, it has not yet actually been used in information retrieval systems. Neverthe-

<sup>1</sup>This work has been partially funded by French National Research Agency (ANR) under project Chronolines (ANR-10-CORD-010). We would like to thank the French News Agency (AFP) for providing us with the corpus.

less, (Alonso et al., 2007; Alonso, 2008; Kanhabua, 2009) and (Mestl et al., 2009), among others, have highlighted that the analysis of temporal information is often an essential component in text understanding and is useful in a wide range of information retrieval applications. (Harabagiu and Bejan, 2005; Saquete et al., 2009) highlight the importance of processing temporal expressions in Question Answering systems. For example, in the TREC-10 QA evaluation campaign, more than 10% of questions required an element of temporal processing in order to be correctly processed (Li et al., 2005a). In multi-document summarization, temporal processing enables a system to detect redundant excerpts from various texts on the same topic and to present results in a relevant chronological order (Barzilay and Elhadad, 2002). Temporal processing is also useful for aiding medical decision-making. (Kim and Choi, 2011) present work on the extraction of temporal information in clinical narrative texts. Similarly, (Jung et al., 2011) present an end-to-end system that processes clinical records, detects events and constructs timelines of patients' medical histories.

The various editions of the TDT task have given rise to the development of different systems that detect novelty in news streams (Allan, 2002; Kumaran and Allen, 2004; Fung et al., 2005). Most of these systems are based on statistical bag-of-words models that use similarity measures to determine proximity between documents (Li et al., 2005b; Brants et al., 2003). (Smith, 2002) used spatio-temporal information from texts to detect events from a digital library. His method used place/time collocations and ranked events according to statistical measures.

Some efforts have been made for automatically building textual and graphical timelines. For example, (Allan et al., 2001) present a system that uses measures of pertinence and novelty to construct timelines that consist of one sentence per date. (Chieu and Lee, 2004) propose a similar system that extracts events relevant to a query from a collection of documents. Important events are those reported in a large number of news articles and each event is constructed according to one single query and represented by a set of sentences. (Swan and Allen, 2000) present an approach to generating graphical timelines that involves extracting clusters of noun phrases and named entities. More recently, (Yan et

al., 2011b; Yan et al., 2011a) used a summarization-based approach to automatically generate timelines, taking into account the evolutionary characteristics of news.

### 3 Resources and System Overview

#### 3.1 AFP Corpus

For this work, we used a corpus of newswire texts provided by the AFP French news agency. The English AFP corpus is composed of 1.3 million texts that span the 2004-2011 period (511 documents/day in average and 426 millions words). Each document is an XML file containing a title, a date of creation (DCT), set of keywords, and textual content split into paragraphs.

#### 3.2 AFP Chronologies

AFP "chronologies" (textual event timelines) are a specific type of articles written by AFP journalists in order to contextualize current events. These chronologies may concern any topic discussed in the media, and consist in a list of dates (typically between 10 and 20) associated with a text describing the related event(s). Figure 1 shows an example of such a chronology. Further examples are given in Figure 2. We selected 91 chronologies satisfying the following constraints:

- All dates in the chronologies are between 2004 and 2011 to be sure that the related events are described in the corpus. For example, "*Chronology of climax to Vietnam War*" was excluded because its corresponding dates do not appear in the content of the articles.
- All dates in the chronology are anterior to the chronology's creation date. For example, the chronology "*Space in 2005: A calendar*", published in January 2005 and listing scheduled events, was not selected (because almost no rocket launches finally happened on the expected day).
- The temporal granularity of the chronology is the day. For example, "*A timeline of how the London transport attacks unfolded*", relating the events hour by hour, is not in our focus.

```

<NewsML Version="1.2">
  <NewsItem xml:lang="en">
    <HeadLine>Key dates in Thailand's political crisis</HeadLine>
    <DateId>20100513T100519Z</DateId>
    <NameLabel>Thailand-politics</NameLabel>
    <DataContent>
      <p>The following is a timeline of events since the protests began, soon after Thailand's Supreme Court confiscated 1.4 billion dollars of Thaksin's wealth for abuse of power.</p>
      <p>March 14: Tens of thousands of Red Shirts demonstrate in the capital calling for Abhisit's government to step down, [...]</p>
      <p>March 28: The government and the Reds enter into talks but hit a stalemate after two days [...]</p>
      <p>April 3: Tens of thousands of protesters move from Bangkok's historic district into the city's commercial heart [...]</p>
      <p>April 7: Abhisit declares state of emergency in capital after Red Shirts storm parliament.</p>
      <p>April 8: Authorities announce arrest warrants for protest leaders.</p>
      ...
    </DataContent>
  </NewsItem>
</NewsML>

```

Figure 1: Example of an AFP manual chronology.

For learning and evaluation purposes, all chronologies were converted to a single XML format. Each document was manually associated with a user search query made up of the keywords required to retrieve the chronology.

### 3.3 System Overview

Figure 3 shows the general architecture of the system. First, pre-processing of the AFP corpus tags and normalizes temporal expressions in each of the articles (step ① in the Figure). Next, the corpus is indexed by the Lucene search engine<sup>2</sup> (step ②).

Given a query, a number of documents are retrieved by Lucene (③). These documents can be filtered (④), and dates are extracted from the remaining documents. These dates are then ranked in order to show the most important ones to the user (⑤), to-

<sup>2</sup><http://lucene.apache.org>

- Chronology of 18 months of trouble in Ivory Coast
- Chechen rebels' history of hostage-takings
- Iraqi political wrangling since March 7 election
- Athletics: Timeline of men's 800m world record
- Major accidents in Chinese mines
- Space in 2005: A calendar
- Developments in Iranian nuclear standoff
- Chronology of climax to Vietnam War
- Timeline of ex-IMF chief's sex attack case
- A timeline of how the London transport attacks unfolded

Figure 2: Examples of AFP chronologies.

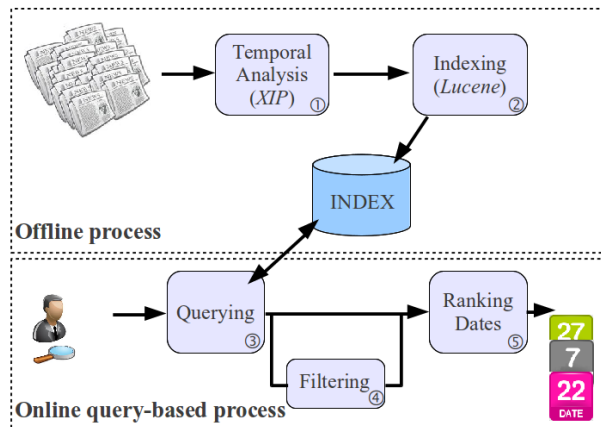


Figure 3: System overview.

gether with the sentences that contain them.

## 4 Temporal and Linguistic Processing

In this section, we describe the linguistic and temporal information extracted during the pre-processing phase and how the extraction is carried out. We rely on the powerful linguistic analyzer XIP (Ait-Mokhtar et al., 2002), that we adapted for our purposes.

### 4.1 XIP

The linguistic analyzer we use performs a deep syntactic analysis of running text. It takes as input XML files and analyzes the textual content enclosed in the various XML tags in different ways that are specified in an XML guide (a file providing instructions to the parser, see (Roux, 2004) for details). XIP performs complete linguistic processing ranging from tokenization to deep grammatical dependency analysis. It also performs named entity recog-



nition (NER) of the most usual named entity categories and recognizes temporal expressions. Linguistic units manipulated by the parser are either terminal categories or chunks. Each of these units is associated with an attribute-value matrix that contains the unit's relevant morphological, syntactic and semantic information. Linguistic constituents are linked by oriented and labelled n-ary relations denoting syntactic or semantic properties of the input text. A Java API is provided with the parser so that all linguistic structures and relations can be easily manipulated by Java code.

In the following subsections, we give details of the linguistic information that is used for the detection of salient dates.

## 4.2 Named Entity Recognition

Named Entity (NE) Recognition is one of the outputs provided by XIP. NEs are represented as unary relations in the parser output. We used the existing NE recognition module of the English grammar which tags the following NE types: **location names**, **person names** and **organization names**. Ambiguous NE types (ambiguity between type **location** or **organization** for country names for instance) are also considered.

## 4.3 Temporal Analysis

A previous module for temporal analysis was developed and integrated into the English grammar (Hagège and Tannier, 2008), and evaluated during TempEval campaign (Verhagen et al., 2007). This module was adapted for tagging salient dates. Our goal with temporal analysis is to be able to tag and normalize<sup>3</sup> a selected subset of temporal expressions (TEs) which we consider to be relevant for our task. This subset of expressions is described in the following sections.

### 4.3.1 Absolute Dates

Absolute dates are dates that can be normalized without external or contextual knowledge. This is the case, for instance, of “*On January 5th 2003*”. In these expressions, all information needed for normalization is contained in the linguistic expression.

<sup>3</sup>We call *normalization* the operation of turning a temporal expression into a formatted, fully specified representation. This includes finding the absolute value of relative dates.

However, absolute dates are relatively infrequent in our corpus (7%), so in order to broaden the coverage for the detection of salient dates, we decided to consider relative dates, which are far more frequent.

### 4.3.2 DCT-relative Dates

DCT-relative temporal expressions are those which are relative to the creation date of the document. This class represents 40% of dates extracted from the AFP corpus. Unlike the absolute dates, the linguistic expression does not provide all the information needed for normalization. External information is required, in particular, the date which corresponds to the moment of utterance. In news articles, this is the DCT. Two sub-classes of relative TEs can be distinguished. The first sub-class only requires knowledge of the DCT value to perform the normalization. This is the case of expressions like *next Friday*, which correspond to the calendar date of the first Friday following the DCT. The second sub-class requires further contextual knowledge for normalization. For example, *on Friday* will correspond either to *last Friday* or to *next Friday* depending on the context where this expression appears (e.g. *He is expected to come on Friday* corresponds to *next Friday* while *He arrived on Friday* corresponds to *last Friday*). In such cases, the tense of the verb that governs the TE is essential for normalization. This information is provided by the linguistic analysis carried out by XIP.

### 4.3.3 Underspecified Dates

Considering the kind of corpus we deal with (news), we decided to consider TEs whose granularity is at least equal to a day. As a result, TEs were normalized to a numerical YYYYMMDD format (where YYYY corresponds to the year, MM to the month and DD to the day). In case of TEs with a granularity superior to the day or month, DD and MM fields remain unspecified accordingly. However, these underspecified dates are not used in our experiments.

## 4.4 Modality and Reported Speech

An important issue that can affect the calculation of salient dates is the modality associated with timestamped events in text. For instance, the status of a salient date candidate in a sentence like “*The meet-*

ing takes place on Friday” has to be distinguished from the one in “The meeting should take place on Friday” or “The meeting will take place on Friday, Mr. Hong said”. The time-stamped event *meeting takes place* is factual in the first example and can be taken as granted. In the second and third examples, however, the event does not necessarily occur. This is expressed by the modality introduced by the modal auxiliary *should* (second example), or by the use of the future tense or reported speech (third example). We annotate TEs with information regarding the factuality of the event they modify. More specifically, we consider the following features:

**Events that are mentioned in the future:** If a time-stamped event is in the future tense, we add a specific attribute MODALITY with value FUTURE to the corresponding TE annotation.

**Events used with a modal verb:** If a time-stamped event is introduced by a modal verb such as *should* or *would*, then attribute MODALITY to the corresponding TE annotation has the value MODAL.

**Reported speech verbs:** Reported speech verbs (or verbs of speaking) introduce indirect or reported speech. We dealt with time-stamped events governed by a reported speech verb, or otherwise appearing in reported speech. Once again, XIP’s linguistic analysis provided the necessary information, including the marking of reported speech verbs and clause segmentation of complex sentences. If a relevant TE modifies a reported speech verb, the annotation of this TE contains a specific attribute, DECLARATION=“YES”. If the relevant TE modifies a verb that appears in a clause introduced by a reported speech verb then the annotation contains the attribute REPORTED=“YES”.

Note that the different annotations can be combined (e.g. modality and reported speech can occur for a same time-stamped event). For example, the TE *Friday* in “The meeting should take place on Friday, Mr. Hong said” is annotated with both modality and reported speech attributes.

#### 4.5 Corpus-dependent Special Cases

While we developed the linguistic and temporal annotators, we took into account some specificities of our corpus. We decided that the TEs *today* and

```
<DCT value="20050105"/>
<EC TYPE="TIMEX" value="unknown">The year
2004</EC> was the deadliest <EC TYPE="TIMEX"
value="unknown">in a decade</EC> for journalists
around the world, mainly because of the number of reporters
killed in <EC TYPE="LOCORG">Iraq</EC>, the
media rights group <EN TYPE="ORG">Reporters
Sans Frontieres</EN> (Reporters Without Bor-
ders) said <EC TYPE="DATE" SUBTYPE="REL"
REF="ST" DECLARATION="YES" value
="20050105">Wednesday</EC>.
```

Figure 4: Example of XIP output for a sample article.

*now* were not relevant for the detection of salient dates. In the AFP news corpus, these expressions are mostly generic expressions synonymous with *nowadays* and do not really time-stamp an event with respect to the DCT. Another specificity of the corpus is the fact that if the DCT corresponds to a Monday, and if an event in a past tense is described with the associated TE *on Monday* or *Monday*, it means that this event occurs on the DCT day itself, and not on the Monday before. We adapted the TE normalizer to these special cases.

#### 4.6 Implementation and Example

As said previously, a NER module is integrated into the XIP parser, which we used “as is”. The TE tagger and normalizer was adapted from (Hagège and Tannier, 2008). We used the Java API provided with the parser to perform the annotation and normalization of TEs. The output for the linguistic and temporal annotation consists in XML files where only selected information is kept (structural information distinguishing headlines from news content, DCT), and enriched with the linguistic annotations described before (NEs and TEs with relevant attributes corresponding to the normalization and typing). Information concerning modality, future tense and reported speech, appears as attributes on the TE tag. Figure 4 shows an example of an analyzed excerpt of a news article.

In this news excerpt, only one TE (*Wednesday*) is normalized as both *The year 2004* and *in a decade* are not considered to be relevant. The first one being a generic TE and the second one being of granularity superior to a year. The annotation of the relevant TE has the attribute indicating that it time-stamps an event realized by a reported speech verb. The nor-

malized value of the TE corresponds to the 5th of January 2005, which is a Wednesday. NEs are also annotated.

In the entire AFP corpus, 11.5 millions temporal expressions were detected, among which 845,000 absolute dates (7%) and 4.6 millions normalized relative dates (40%). Although we have not yet evaluated our tagging of relative dates, the system on which our current date normalization is based achieved good results in the TempEval (Verhagen et al., 2007) campaign.

## 5 Experiments and Results

In Section 5.1, we propose two baseline approaches in order to give a good idea of the difficulty of the task (Section 5.4 also discusses this point). In Section 5.2, we present our experiments using simple filtering and statistics on dates calculated by Lucene. Finally, Section 5.3 gives details of our experiments with a learning approach. In our experiments, we used three different values to rank dates:

- $occ(d)$  is the number of textual units (documents or sentences) containing the date  $d$ .
- Lucene provides ranked documents together with their relevance score.  $luc(d)$  is the sum of Lucene scores for textual units containing the date  $d$ .
- An adaptation of classical  $tf.idf$  for dates:

$$tf.idf(d) = f(d) \cdot \log \frac{N}{df(d)}$$

where  $f(d)$  is the number of occurrences of date  $d$  in the sentence (generally,  $f(d) = 1$ ),  $N$  is the number of indexed sentences and  $df(d)$  is the number of sentences containing date  $d$ .

In all experiments (including baselines), timelines have been built by considering only dates between the first and the last dates of the corresponding manual chronology. Processing runs were evaluated on manually-written chronologies (see Section 3.2) according to Mean Average Precision (MAP), which is a widely accepted metric for ranked lists. MAP gives a higher weight to higher ranked elements than lower ranked elements. Significance of evaluation results are indicated by the  $p$ -value results of the Student’s t-test ( $t(90) = 1.9867$ ).

Baselines “only DCTs”			
Model	$BL_{DCT}^{occ}$	$BL_{DCT}^{luc}$	$BL_{DCT}^{tf.idf}$
MAP Score	0.5036	0.5521	0.5523
Baselines “only absolute dates”			
Model	$BL_{abs}^{occ}$	$BL_{abs}^{luc}$	$BL_{abs}^{tf.idf}$
MAP Score	0.2627	0.2782	0.2778
Baselines “absolute dates or alternatively DCTs”			
Model	$BL_{mix}^{occ}$	$BL_{mix}^{luc}$	$BL_{mix}^{tf.idf}$
MAP Score	0.4005	0.4110	0.4135

Table 1: MAP results for baseline runs.

### 5.1 Baseline Runs

$BL_{DCT}$ . Indexing and search were done at **document level** (*i.e.* each AFP article, with its title and keywords, is a document). Given a query, the top 10,000 documents were retrieved. In these runs, **only the DCT** for each document was considered. Dates were ranked by one of the three values described above ( $occ$ ,  $luc$  or  $tf.idf$ ) leading to runs  $BL_{DCT}^{occ}$ ,  $BL_{DCT}^{luc}$  and  $BL_{DCT}^{tf.idf}$ .

$BL_{abs}$ . Indexing and search were done at **sentence level** (document title and keywords are added to sentence text). Given a query, the top 10,000 sentences were retrieved. **Only absolute dates** in these sentences were considered. We thus obtained runs  $BL_{abs}^{occ}$ ,  $BL_{abs}^{luc}$  and  $BL_{abs}^{tf.idf}$ .

Note that in this baseline, as well as in all the subsequent runs, the information unit was the sentence because a date was associated to a small part of the text. The rest of the document generally contained text that was not related to the specific date.

$BL_{mix}$ . Same as  $BL_{abs}$ , except that sentences containing no absolute dates were considered and associated to the DCT.

Table 1 shows results for these baseline runs. Using only DCTs with Lucene scores or  $tf.idf(d)$  already yielded interesting results, with MAP around 0.55.

### 5.2 Salient Date Extraction with XIP Results and Simple Filtering

In these experiments, we considered a Lucene index to be built as follows: each document was taken to

Model	MAP Score	Model	MAP Score
<i>Salient date runs with all dates</i>			
$SD^{luc}$	0.6962	$SD^{tf.idf}$	0.6982
<i>Salient dates runs with filtering</i>			
$SD_R^{luc}$	0.6975	$SD_R^{tf.idf}$	0.6996
$SD_F^{luc}$	0.6967	$SD_F^{tf.idf}$	0.6993 **
$SD_M^{luc}$	0.6978	$SD_M^{tf.idf}$	0.7005 *
$SD_D^{luc}$	0.7066 **	$SD_D^{tf.idf}$	0.7091 **
$SD_{FMD}^{luc}$	0.7086 **	$SD_{FMD}^{tf.idf}$	0.7112 **
$SD_{RFMD}^{luc}$	0.7127 **	$SD_{RFMD}^{tf.idf}$	0.7146 **

Table 2: MAP results for salient date extraction with XIP and simple filtering. The significance of the improvement due to filtering wrt no filtering is indicated by the Student t-test (\*:  $p < 0.05$  (significant); \*\*:  $p < 0.01$  (highly significant)). The improvement due to using  $tf.idf(d)$  as opposed to  $occ(d)$  is also highly significant.

be a **sentence** containing a **normalized date**. This sentence was indexed with the title and keywords of the AFP article containing it. Given a query, the top 10,000 documents were retrieved. Combinations between the following filtering operations were possible, by removing all dates associated with a reported speech verb ( $R$ ), a modal verb ( $M$ ) and/or a future verb ( $F$ ). All these filtering operations were intended to remove references to events that were not certain, thereby minimizing noise in results.

These processing runs are named  $SD$  runs, with indices representing the filtering operations. For example, a run obtained by filtering modal and future verbs is called  $SD_{M,F}$ . In all combinations, dates were ranked by the sum of Lucene scores for these sentences ( $luc$ ) or by  $tf.idf^4$ .

Table 2 presents the results for this series of experiments. MAP values are much higher than for baselines. Using  $tf.idf(d)$  is only very slightly better than  $luc$ . Filtering operations bring significant improvement but the benefits of these different techniques have to be further investigated.

### 5.3 Machine-Learning Runs

We used our set of manually-written chronologies as a training corpus to perform machine learning experiments. We used IcsiBoost<sup>5</sup>, an implementa-

<sup>4</sup>We do not present runs where dates are ranked by the number of times they appear in retrieved sentences ( $occ$ ), as we did for baselines, since results are systematically lower.

<sup>5</sup><http://code.google.com/p/icsiboost/>

tion of adaptative boosting (AdaBoost (Freund and Schapire, 1997)).

In our approach, we consider two classes: *salient dates* are dates that have an entry in the manual chronologies, while *non-salient dates* are all other dates. This choice does, however, represent an important bias. The choices of journalists are indeed very subjective, and chronologies must not exceed a certain length, which means that relevant dates can be thrown away. These issues will be discussed in Section 5.4.

The classifier instances were not all sentences retrieved by the search engine. Using all sentences would not yield a useful feature set. We rather aggregated all sentences corresponding to the same date before learning the classifier. Therefore, each instance corresponded to a single date, and features were figures concerning the set of sentences containing this date.

Features used in this series of runs are as follows:

1. Features representing the fact that the more a date is mentioned, the more important it is likely to be: 1) Sum of the Lucene scores for all sentences containing the date 2) Number of sentences containing the date 3) Ratio between the total weights of the date and weights of all returned dates 4) Ratio between the frequency of the date and frequency of all returned dates;
2. Features representing the fact that an important event is still written about, a long time after it occurs: 1) Distance between the date and the most recent mention of this date 2) Distance between the date and the DCT;
3. Other features: 1) Lucene's best ranking of the date 2) Number of times where the date is absolute in the text 3) Number of times where the date is relative (but normalized) in the text 4) Total number of keywords of the query in the title, sentence and named entities of retrieved documents 5) Number of times where the date modifies a reported speech verb or is extracted from reported speech.

We did not aim to classify dates, but rather to rank them. Instead, we used the predicted probability  $P(d)$  returned by the classifier, and mixed it with the Lucene score of sentences, or with date  $tf.idf$ :

Model	MAP Score
<i>Machine-Learning Runs</i>	
$ML_{base}^{luc}$	0.7033
$ML^{luc}$	0.7905 **
$ML^{tf.idf}$	0.7918 **

Table 3: MAP results for salient date extraction with machine-learning.  $ML_{base}^{luc}$  used Lucene scores and only the first set of features described above.  $ML^{luc}$  and  $ML^{tf.idf}$  used the three sets of features. They are both highly significant under the *t-test* ( $p \approx 6.10^{-4}$ ) wrt respectively  $SD^{luc}$  and  $SD^{tf.idf}$ .

$$score(d) = P(d) \times val(d)$$

where  $val(d)$  is either  $luc(d)$  or  $tf.idf(d)$ .

Because the task is very subjective and (above all) because of the low quantity of learning data, we preferred not to opt for a “learning to rank” approach.

We evaluated this approach with a classic 4-fold cross-validation. Our 91 chronologies were randomly divided into 4 sub-samples, each of them being used once as test data. The final scores, presented in Table 3, are the average of these 4 processes. As shown in this table, the learning approach improves MAP results by about 0.05 point.

#### 5.4 Discussion and Final Experiment

Chronologies hand-written by journalists are a very useful resources for evaluation of our system, as they are completely dissociated from our research and are an exact representation of the output we aim to obtain. However, assembling such a chronology is a very subjective task, and no clear method for evaluation agreement between two journalists seems immediately apparent. Only experts can build such chronologies, and calculating this agreement would require at least two experts from each domain, which are hard to come by. One may then consider our system as a useful tool for building a chronology more objectively.

To illustrate this point, we chose four specific topics<sup>6</sup> and showed one of our runs on each topic to an AFP expert for these subjects. We asked him to assess the first 30 dates of these runs.

<sup>6</sup>Namely, “Arab revolt timeline for Morocco”, “Kyrgyzstan unrest timeline”, “Lebanon’s new government: a timeline”, “Libya timeline”.

Topic	$AP_C$	$AP_E$
<i>Morocco</i>	0.5847	0.5718
<i>Kyrgyzstan</i>	0.6125	0.9989
<i>Libya</i>	0.7856	1
<i>Lebanon</i>	0.4673	0.7652

Table 4: Average precision results for manual evaluation on 4 topics, against the original chronologies ( $AP_C$ ), and the expert assessment ( $AP_E$ ).

Table 4 presents results for this evaluation, comparing average precision values obtained 1) against the original, manual chronologies ( $AP_C$ ), and 2) against the expert assessment ( $AP_E$ ). These values show that, for 3 runs out of 4, many dates returned by the system are considered as valid by the expert, even if not presented in the original chronology.

Even if this experiment is not strong enough to lead to a formal conclusion (*post-hoc* evaluation with only 4 topics and a single assessor), this tends to show that our system produces usable outputs and that our system can be of help to journalists by providing them with chronologies that are as useful and objective as possible.

## 6 Conclusion and Future Work

This article presents a task of “date extraction” and shows the importance of taking temporal information into consideration and how with relatively simple temporal processing, we were able to indirectly point to important events using the temporal information associated with these events. Of course, as our final goal consists in the detection of important events, we need to take into account the textual content. In future work, we envisage providing, together with the detection of salient dates, a semantic analysis that will help determine the importance of events. Another interesting direction in which we soon aim to work is to consider all textual excerpts that are associated with salient dates, and use clustering techniques to determine if textual excerpts correspond to the same event or not. Finally, as our news corpus is available both for English and French (comparable corpus, not necessarily translations), we aim to investigate cross-lingual extraction of salient dates and salient events.

## References

- Salah Aït-Mokhtar, Jean-Pierre Chanod, and Claude Roux. 2002. Robustness beyond Shallowness: Incremental Deep Parsing. *Natural Language Engineering*, 8:121–144.
- James Allan, Rahul Gupta, and Vikas Khandelwal. 2001. Temporal summaries of new topics. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 10–18.
- James Allan, editor. 2002. *Topic Detection and Tracking*. Springer.
- Omar Alonso, Ricardo Baeza-Yates, and Michael Gertz. 2007. Exploratory Search Using Timelines. In *SIGCHI 2007 Workshop on Exploratory Search and HCI Workshop*.
- Omar Rogelio Alonso. 2008. *Temporal information retrieval*. Ph.D. thesis, University of California at Davis, Davis, CA, USA. Adviser-Gertz, Michael.
- Regina Barzilay and Noemie Elhadad. 2002. Inferring Strategies for Sentence Ordering in Multidocument News Summarization. *Journal of Artificial Intelligence Research*, 17:35–55.
- Thorsten Brants, Francine Chen, and Ayman Farahat. 2003. A system for new event detection. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '03, pages 330–337, New York, NY, USA. ACM.
- Hai Leong Chieu and Yoong Keok Lee. 2004. Query based event extraction along a timeline. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 425–432.
- Yoav Freund and Robert E. Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139.
- Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Philip S. Yu, and Hongjun Lu. 2005. Parameter free bursty events detection in text streams. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 181–192.
- Caroline Hagège and Xavier Tannier. 2008. XTM: A Robust Temporal Text Processor. In *Computational Linguistics and Intelligent Text Processing, proceedings of 9th International Conference CICLing 2008*, pages 231–240, Haifa, Israel, February. Springer Berlin / Heidelberg.
- Sanda Harabagiu and Cosmin Adrian Bejan. 2005. Question Answering Based on Temporal Inference. In *Proceedings of the Workshop on Inference for Textual Question Answering*, Pittsburg, Pennsylvania, USA, July.
- Hyuckchul Jung, James Allen, Nate Blaylock, Will de Beaumont, Lucian Galescu, and Mary Swift. 2011. Building timelines from narrative clinical records: initial results based-on deep natural language understanding. In *Proceedings of BioNLP 2011 Workshop*, BioNLP '11, pages 146–154, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Nattiya Kanhabua. 2009. Exploiting temporal information in retrieval of archived documents. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR 2009, Boston, MA, USA, July 19–23, 2009, page 848.
- Youngho Kim and Jinwook Choi. 2011. Recognizing temporal information in korean clinical narratives through text normalization. *Health Inform Res*, 17(3):150–5.
- Giridhar Kumaran and James Allen. 2004. Text classification and named entities for new event detection. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 297–304. ACM.
- Wei Li, Wenjie Li, Qin Lu, and Kam-Fai Wong. 2005a. A Preliminary Work on Classifying Time Granularities of Temporal Questions. In *Proceedings of Second international joint conference in NLP (IJCNLP 2005)*, Jeju Island, Korea, oct.
- Zhiwei Li, Bin Wang, Mingjing Li, and Wei-Ying Ma. 2005b. A Probabilistic Model for Restrospective News Event Detection. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Salvador, Brazil. ACM Press, New York City, NY, USA.
- Thomas Mestl, Olga Cerrato, Jon Ølnes, Per Myrseth, and Inger-Mette Gustavsen. 2009. Time Challenges - Challenging Times for Future Information Search. *D-Lib Magazine*, 15(5/6).
- James Pustejovsky, Kiyong Lee, Harry Bunt, and Laurent Romary. 2010. Iso-timeml: An international standard for semantic annotation. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, may. European Language Resources Association (ELRA).
- Claude Roux. 2004. Annoter les documents XML avec un outil d'analyse syntaxique. In *11ème Confrence annuelle de Traitement Automatique des Langues Naturelles*, Fès, Morocco, April. ATALA.

- Estela Saquete, Jose L. Vicedo, Patricio Martínez-Barco, Rafael Muñoz, and Hector Llorens. 2009. Enhancing QA Systems with Complex Temporal Question Processing Capabilities. *Journal of Artificial Intelligence Research*, 35:775–811.
- David A. Smith. 2002. Detecting events with date and place information in unstructured text. In *JCDL '02: Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*, pages 191–196, New York, NY, USA. ACM.
- Russell Swan and James Allen. 2000. Automatic generation of overview timelines. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '00*, pages 49–56, New York, NY, USA. ACM.
- Marc Verhagen, Robert Gaizauskas, Franck Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. SemEval-2007 - 15: TempEval Temporal Relation Identification. In *Proceedings of SemEval workshop at ACL 2007*, Prague, Czech Republic, June. Association for Computational Linguistics, Morristown, NJ, USA.
- Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. 2011a. Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, Edinburgh, UK*, pages 433–443.
- Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. 2011b. Evolutionary timeline summarization: a balanced optimization framework via iterative substitution. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*, pages 745–754.
- Y. Yang, T. Pierce, and J. G. Carbonell. 1998. A study on retrospective and on-line event detection. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Melbourne, Australia, August. ACM Press, New York City, NY, USA.

# Historical Analysis of Legal Opinions with a Sparse Mixed-Effects Latent Variable Model

William Yang Wang<sup>1</sup> and Elijah Mayfield<sup>1</sup> and Suresh Naidu<sup>2</sup> and Jeremiah Dittmar<sup>3</sup>

<sup>1</sup>School of Computer Science, Carnegie Mellon University

<sup>2</sup>Department of Economics and SIPA, Columbia University

<sup>3</sup>American University and School of Social Science, Institute for Advanced Study

{ww,elijah}@cmu.edu sn2430@columbia.edu dittmar@american.edu

## Abstract

We propose a latent variable model to enhance historical analysis of large corpora. This work extends prior work in topic modelling by incorporating metadata, and the interactions between the components in metadata, in a general way. To test this, we collect a corpus of slavery-related United States property law judgements sampled from the years 1730 to 1866. We study the language use in these legal cases, with a special focus on shifts in opinions on controversial topics across different regions. Because this is a longitudinal data set, we are also interested in understanding how these opinions change over the course of decades. We show that the joint learning scheme of our sparse mixed-effects model improves on other state-of-the-art generative and discriminative models on the region and time period identification tasks. Experiments show that our sparse mixed-effects model is more accurate quantitatively and qualitatively interesting, and that these improvements are robust across different parameter settings.

## 1 Introduction

Many scientific subjects, such as psychology, learning sciences, and biology, have adopted computational approaches to discover latent patterns in large scale datasets (Chen and Lombardi, 2010; Baker and Yacef, 2009). In contrast, the primary methods for historical research still rely on individual judgement and reading primary and secondary sources, which are time consuming and expensive. Furthermore, traditional human-based methods might have good precision when searching for relevant information, but suffer from low recall. Even when language technologies have been applied to historical problems, their focus has often been on information retrieval (Gotscharek et al., 2009), to improve accessibility of texts. Empirical methods for analysis and interpretation of these texts is therefore a burgeoning new field.

Court opinions form one of the most important parts of the legal domain, and can serve as an excellent resource to understand both legal and political history (Popkin, 2007). Historians often use court opinions as a primary source for constructing interpretations of the past. They not only report the proceedings of a court, but also express a judges' views toward the issues at hand in a case, and reflect the legal and political environment of the region and period. Since there exists many thousands of early court opinions, however, it is difficult for legal historians to manually analyze the documents case by case. Instead, historians often restrict themselves to discussing a relatively small subset of legal opinions that are considered decisive. While this approach has merit, new technologies should allow extraction of patterns from large samples of opinions.

Latent variable models, such as latent Dirichlet allocation (LDA) (Blei et al., 2003) and probabilistic latent semantic analysis (PLSA) (Hofmann, 1999), have been used in the past to facilitate social science research. However, they have numerous drawbacks, as many topics are uninterpretable, overwhelmed by uninformative words, or represent background language use that is unrelated to the dimensions of analysis that qualitative researchers are interested in.

SAGE (Eisenstein et al., 2011a), a recently proposed sparse additive generative model of language, addresses many of the drawbacks of LDA. SAGE assumes a background distribution of language use, and enforces sparsity in individual topics. Another advantage, from a social science perspective, is that SAGE can be derived from a standard logit random-utility model of judicial opinion writing, in contrast to LDA. In this work we extend SAGE to the supervised case of joint region and time period prediction. We formulate the resulting sparse mixed-effects (SME) model as being made up of mixed effects that not only contain random effects from sparse topics, but also mixed effects from available metadata. To do this we augment SAGE with two sparse latent variables that model the region and time of a document, as well as a third sparse latent



variable that captures the interactions among the region, time and topic latent variables. We also introduce a multiclass perceptron-style weight estimation method to model the contributions from different sparse latent variables to the word posterior probabilities in this predictive task. Importantly, the resulting distributions are still sparse and can therefore be qualitatively analyzed by experts with relatively little noise.

In the next two sections, we overview work related to qualitative social science analysis using latent variable models, and introduce our slavery-related early United States court opinion data. We describe our sparse mixed-effects model for joint modeling of region, time, and topic in section 4. Experiments are presented in section 5, with a robust analysis from qualitative and quantitative standpoints in section 5.2, and we discuss the conclusions of this work in section 6.

## 2 Related Work

Natural Language Processing (NLP) methods for automatically understanding and identifying key information in historical data have not yet been explored until recently. Related research efforts include using the LDA model for topic modeling in historical newspapers (Yang et al., 2011), a rule-based approach to extract verbs in historical Swedish texts (Pettersson and Nivre, 2011), a system for semantic tagging of historical Dutch archives (Cybulska and Vossen, 2011).

Despite our historical data domain, our approach is more relevant to text classification and topic modelling. Traditional discriminative methods, such as support vector machine (SVM) and logistic regression, have been very popular in various text categorization tasks (Joachims, 1998; Wang and McKeown, 2010) in the past decades. However, the main problem with these methods is that although they are accurate in classifying documents, they do not aim at helping us to understand the documents.

Another problem is lack of expressiveness. For example, SVM does not have latent variables to model the subtle differences and interactions of features from different domains (e.g. text, links, and date), but rather treats them as a “bag-of-features”. Generative methods, by contrast, can show the causes to effects, have attracted attentions in recent years due to the rich expressiveness of the models and competitive performances in predictive tasks (Wang et al., 2011). For example, Nguyen et al. (2010) study the effect of the context of interaction in blogs using a standard LDA model. Guo and Diab (2011) show the effectiveness of using se-

mantic information in multifaceted topic models for text categorization. Eisenstein et al. (2010) use a latent variable model to predict geolocation information of Twitter users, and investigate geographic variations of language use. Temporally, topic models have been used to show the shift in language use over time in online communities (Nguyen and Rosé, 2011) and the evolution of topics over time (Shubhankar et al., 2011).

When evaluating understandability, however, dense word distributions are a serious issue in many topic models as well as other predictive tasks. Such topic models are often dominated by function words and do not always effectively separate topics. Recent work have shown significant gains in both predictiveness and interpretability by enforcing sparsity, such as in the task of discovering sociolinguistic patterns of language use (Eisenstein et al., 2011b).

Our proposed sparse mixed-effects model balances the pros and cons the above methods, aiming at higher classification accuracies using the SME model for joint geographic and temporal aspects prediction, as well as richer interaction of components from metadata to enhance historical analysis in legal opinions. To the best of our knowledge, this study is the first of its kind to discover region and time specific topical patterns jointly in historical texts.

## 3 Data

We have collected a corpus of slavery-related United States supreme court legal opinions from Lexis Nexis. The dataset includes 5,240 slavery-related state supreme court cases from 24 states, during the period of 1730 - 1866. Optical character recognition (OCR) software was used by Lexis Nexis to digitize the original documents. In our region identification task, we wish to identify whether an opinion was written in a free state<sup>1</sup> (R1) or a slave state (R2)<sup>2</sup>. In our time identification experiment, we approximately divide the legal documents into four time quartiles (Q1, Q2, Q3, and Q4), and predict which quartile the testing document belongs to. Q1 contains cases from 1837 or earlier, where as Q2 is for 1838-1848, Q3 is for 1849-1855, and Q4 is for 1856 and later.

## 4 The Sparse Mixed-Effects Model

To address the over-parameterization, lack of expressiveness and robustness issues in LDA, the SAGE (Eisenstein et al., 2011a) framework draws a

<sup>1</sup>Including border states, this set includes CT, DE, IL, KY, MA, MD, ME, MI, NH, NJ, NY, OH, PA, and RI.

<sup>2</sup>These states include AR, AL, FL, GA, MS, NC, TN, TX, and VA.



- For each document  $d$ 
  - Draw the region label  $y_d^{(R)}$
  - Draw the time quartile label  $y_d^{(Q)}$
  - For each word  $n$ , draw  $w_n^{(d)} \sim \beta_{y_d}$

#### 4.1 Parameter Estimation

We follow the MAP estimation method that Eisenstein et al. (2011a) used to train all sparse latent variables  $\eta$ , and perform Bayesian inference on other latent variables. The estimation of all variance variables  $\tau$  remains as plugging the compound distribution of Normal-Jeffrey’s prior, where the latter is a replacement of the Exponential prior. When performing Expectation-Maximization (EM) algorithm to infer the latent variables in SME, we derive the following likelihood function:

$$\begin{aligned}
\mathcal{L} = & \sum_d \langle \log P(\theta_d | \alpha) \rangle + \langle \log P(Z_n^{(d)} | \theta_d) \rangle \\
& + \sum_n^{N_d} \langle \log P(w_n^{(d)} | z_n^{(d)}, \eta, m, y_d^{(R)}, y_d^{(Q)}) \rangle \\
& + \sum_k \langle \log P(\eta_k^{(T)} | 0, \tau_k^{(T)}) \rangle + \sum_k \langle \log P(\tau_k^{(T)} | \gamma) \rangle \\
& + \sum_j \langle \log P(\eta_j^{(R)} | 0, \tau_j^{(R)}) \rangle + \sum_j \langle \log P(\tau_j^{(R)} | \gamma) \rangle \\
& + \sum_q \langle \log P(\eta_q^{(Q)} | 0, \tau_q^{(Q)}) \rangle + \sum_q \langle \log P(\tau_q^{(Q)} | \gamma) \rangle \\
& + \sum_q \sum_j \sum_k \langle \log P(\eta_{q,j,k}^{(I)} | 0, \tau_{q,j,k}^{(I)}) \rangle \\
& + \sum_q \sum_j \sum_k \langle \log P(\tau_{q,j,k}^{(I)} | \gamma) \rangle \\
& - \langle \log Q(\tau, z, \theta) \rangle
\end{aligned}$$

The above E step likelihood score can be intuitively interpreted as the sum of topic proportion scores, latent topic scores, the word scores, the  $\eta$  scores with their priors, and minus the joint variance. In the M step, when we use Newton’s method to optimize the sparse deviation  $\eta_k$  parameter, we need to modify the original likelihood function in SAGE and its corresponding first and second order derivatives when deriving the gradient and Hessian matrix. The likelihood function for sparse topic deviation  $\eta_k$  is:

$$\begin{aligned}
\mathcal{L}(\eta_k) = & \langle c_k^{(T)} \rangle \top \eta_k \\
& - C_d \log \sum_q \sum_j \sum_i \exp(\lambda^{(Q)} \eta_{qi} + \lambda^{(R)} \eta_{ji}) \\
& + \eta_{ki} + \eta_{qjk i} + m_i - \eta_k \top \text{diag}(\langle (\tau_k^{(T)})^{-1} \rangle) \eta_k^{(T)} / 2
\end{aligned}$$

and we can derive the gradient when taking the first order partial derivative:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \eta_k^{(T)}} = & \langle c_k^{(T)} \rangle - \sum_q \sum_j \langle C_{qjk} \rangle \beta_{qjk} \\
& - \text{diag}(\langle (\tau_k^{(T)})^{-1} \rangle) \eta_k^{(T)}
\end{aligned}$$

where  $c_k^{(T)}$  is the true count, and  $\beta_{qjk}$  is the log word likelihood in the original likelihood function.  $C_{qjk}$  is the expected count from combinations of time, region and topic.  $\sum_q \sum_j \langle C_{qjk} \rangle \beta_{qjk}$  will then be taken the second order derivative to form the Hessian matrix, instead of  $\langle C_k \rangle \beta_k$  in the previous SAGE setting.

To learn the weight parameters  $\lambda^{(R)}$  and  $\lambda^{(Q)}$ , we can approximate the weights using a multiclass perceptron-style (Collins, 2002) learning method. If we say that the notation of  $\sum V^{(\bar{R})}$  is to marginalize out all other variables in  $\beta$  except  $\eta^{(R)}$ , and  $P(y_d^{(R)})$  is the prior for the region prediction task, we can predict the expected region value  $\hat{y}_d^{(R)}$  of a document  $d$ :

$$\begin{aligned}
\hat{y}_d^{(R)} & \propto \arg \max_{\hat{y}_d^{(R)}} \exp \left( \sum V^{(\bar{R})} \log \beta + \log P(y_d^{(R)}) \right) \\
= \arg \max_{\hat{y}_d^{(R)}} & \left( \exp \left( \sum V^{(\bar{R})} (m + \eta_{z_n^{(d)}}^{(T)} + \lambda^{(R)} \eta_{y_d^{(R)}}^{(R)} \right. \right. \\
& \left. \left. + \lambda^{(Q)} \eta_{y_d^{(Q)}}^{(Q)} + \eta_{y_d^{(R)}, y_d^{(Q)}, z_n^{(d)}}^{(I)} \right) P(y_d^{(R)}) \right)
\end{aligned}$$

If the symbol  $\delta$  is the hyperprior for the learning rate and  $\hat{y}_d^{(R)}$  is the true label, the update procedure for the weights becomes:

$$\lambda_d^{(R')} = \lambda_d^{(R)} + \delta (\hat{y}_d^{(R)} - y_d^{(R)})$$

Similarly, we derive the  $\lambda^{(Q)}$  parameter using the above formula. It is necessary to normalize the weights in each EM loop to preserve the sparsity property of latent variables. The weight update of  $\lambda^{(R)}$  and  $\lambda^{(Q)}$  is bound by the averaged accuracy of the two classification tasks in the training data, which is similar to the notion of minimizing empirical risk (Bahl et al., 1988). Our goal is to choose the two weight parameters that minimize the empirical classification error rate on training data when learning the word posterior probability.

#### 5 Prediction Experiments

We perform three quantitative experiments to evaluate the predictive power of the sparse mixed-effects model. In these experiments, to predict the region and time period labels of a given document, we

jointly learn the two labels in the SME model, and choose the pair which maximizes the probability of the document.

In the first experiment, we compare the prediction accuracy of our SME model to a widely used discriminative learner in NLP – the linear kernel support vector machine (SVM)<sup>3</sup>. In the second experiment, in addition to the linear kernel SVM, we also compare our SME model to a state-of-the-art sparse generative model of text (Eisenstein et al., 2011a), and vary the size of input vocabulary  $W$  exponentially from  $2^9$  to the full size of our training vocabulary<sup>4</sup>. In the third experiment, we examine the robustness of our model by examining how the number of topics influences the prediction accuracy when varying the  $K$  from 10 to 50.

Our data consists of 4615 training documents and 625 held-out documents for testing. While individual judges wrote multiple opinions in our corpus, no judges overlapped between training and test sets. When measuring by the majority class in the testing condition, the chance baseline for the region identification task is 57.1% and the time identification task is 32.3%. We use three-fold cross-validation to infer the learning rate  $\delta$  and cost  $C$  hyperpriors in the SME and SVM model respectively. We use the paired student  $t$ -test to measure the statistical significance.

## 5.1 Quantitative Results

### 5.1.1 Comparing SME to SVM

We show in this section the predictive power of our sparse mixed-effects model, comparing to a linear kernel SVM learner. To compare the two models in different settings, we first empirically set the number of topics  $K$  in our SME model to be 25, as this setting was shown to yield a promising result in a previous study (Eisenstein et al., 2011a) on sparse topic models. In terms of the size of vocabulary  $W$  for both the SME and SVM learner, we select three values to represent dense, medium or sparse feature spaces:  $W_1 = 2^9$ ,  $W_2 = 2^{12}$ , and the full vocabulary size of  $W_3 = 2^{13.8}$ . Table 1 shows the accuracy of both models, as well as the relative improvement (gain) of SME over SVM.

When looking at the experiment results under different settings, we see that the SME model always outperforms the SVM learner. In the time quartile prediction task, the advantage of SME model

<sup>3</sup>In our implementation, we use LibSVM (Chang and Lin, 2011).

<sup>4</sup>To select the vocabulary size  $W$ , we rank the vocabulary by word frequencies in a descending order, and pick the top- $W$  words.

Method	Time	Gain	Region	Gain
SVM ( $W_1$ )	33.2%	–	69.7%	–
SME ( $W_1$ )	36.4%	9.6%	71.4%	2.4%
SVM ( $W_2$ )	35.8%	–	72.3%	–
SME ( $W_2$ )	40.9%	14.2%	74.0%	2.4%
SVM ( $W_3$ )	36.1%	–	73.5%	–
SME ( $W_3$ )	41.9%	16.1%	74.8%	1.8%

Table 1: Compare the accuracy of the linear kernel support vector machine to our sparse mixed-effects model in the region and time identification tasks ( $K = 25$ ). *Gain: the relative improvement of SME over SVM.*

is more salient. For example, with a medium density feature space of  $2^{12}$ , SVM obtained an accuracy of 35.8%, but SME achieved an accuracy of 40.9%, which is a 14.2% relative improvement ( $p < 0.001$ ) over SVM. When the feature space becomes sparser, the SME obtains an increased relative improvement ( $p < 0.001$ ) of 16.1%, using full size of vocabulary. The performance of SVM in the binary region classification is stronger than in the previous task, but SME is able to outperform SVM in all three settings, with tightened advantages ( $p < 0.05$  in  $W_2$  and  $p < 0.001$  in  $W_3$ ). We hypothesize that it might be because that SVM, as a strong large margin learner, is a more natural approach in a binary classification setting, but might not be the best choice in a four-way or multiclass classification task.

### 5.1.2 Comparing SME to SAGE

In this experiment, we compare SME with a state-of-the-art sparse generative model: SAGE (Eisenstein et al., 2011a).

Most studies on topic modelling have not been able to report results when using different sizes of vocabulary for training. Because of the importance of interpretability for social science research, the choice of vocabulary size is critical to ensure understandable topics. Thus we report our results at various vocabulary sizes  $W$  on SME and SAGE. To better validate the performance of SME, we also include the performance of SVM in this experiment, and fix the number of topics  $K = 10$  for the SME and SAGE models, which is a different value for the number of topics  $K$  than the empirical  $K$  we used in the experiment of Section 5.1.1. Figure 2 and Figure 3 show the experiment results in both time and region classification task.

In Figure 2, we evaluate the impacts of  $W$  on our time quartile prediction task. The advantage of the SME model is very obvious throughout the experiments. Interestingly, when we continue to increase

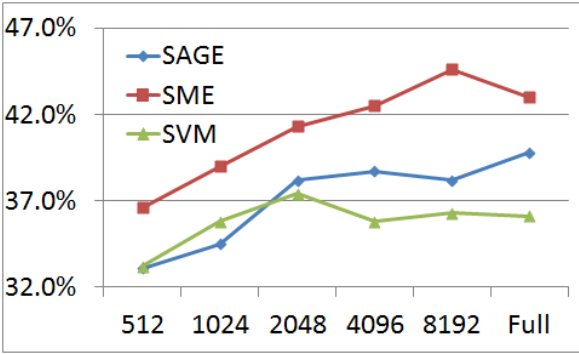


Figure 2: Accuracy on predicting the time quartile varying the vocabulary size  $W$ , while  $K$  is fixed to 10.

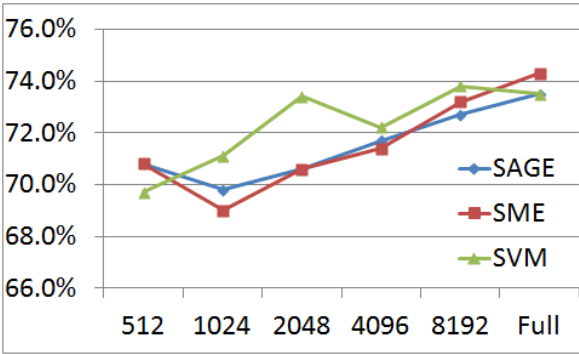


Figure 3: Accuracy on predicting the region varying the vocabulary size  $W$ , while  $K$  is fixed to 10.

the vocabulary size  $W$  exponentially and make the feature space more sparse, SME obtains its best result at  $W = 2^{13}$ , where the relative improvement over SAGE and SVM is 16.8% and 22.9% respectively ( $p < 0.001$  under all comparisons).

Figure 3 shows the impacts of  $W$  on the accuracy of SAGE and SME in the region identification task. In this experiment, the results of SME model are in line with SAGE and SVM when the feature space is dense. However, when  $W$  reaches the full vocabulary size, we have observed significantly better results ( $p < 0.001$  in the comparison to SAGE and  $p < 0.05$  with SVM). We hypothesize that there might be two reasons: first, the  $K$  parameter is set to 10 in this experiment, which is much denser than the experiment setting in Section 5.1.1. Under this condition, the sparse topic advantage of SME might be less salient. Secondly, in the two tasks, it is observed that the accuracy of the binary region classification task is much higher than the four-way task, thus while the latter benefits significantly from this joint learning scheme of the SME model, but the former might not have the equivalent gain<sup>5</sup>.

<sup>5</sup>We hypothesize that this problem might be eliminated if

### 5.1.3 Influence of the number of topics $K$

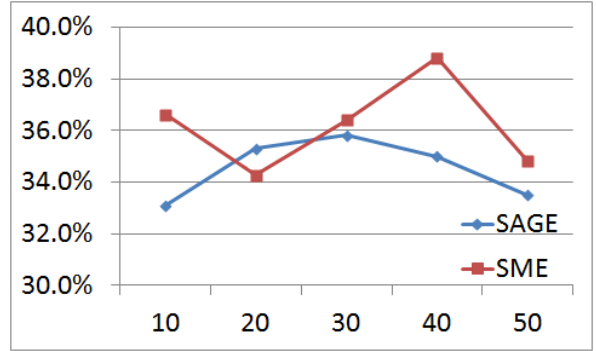


Figure 4: Accuracy on predicting the time quartile varying the number of topics  $K$ , while  $W$  is fixed to  $2^9$ .

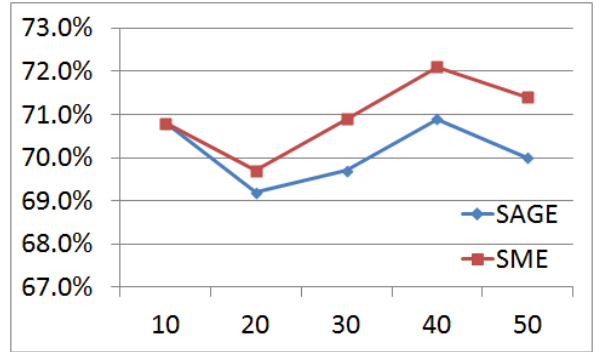


Figure 5: Accuracy on predicting the region varying the number of topics  $K$ , while  $W$  is fixed to  $2^9$ .

Unlike hierarchical Dirichlet processes (Teh et al., 2006), in parametric Bayesian generative models, the number of topics  $K$  is often set manually, and can influence the model’s accuracy significantly. In this experiment, we fix the input vocabulary  $W$  to  $2^9$ , and compare the mixed-effect model with SAGE in both region and time identification tasks.

Figure 4 shows how the variations of  $K$  can influence the system performance in the time quartile prediction task. We can see that the sparse mixed-effects model (SME) reaches its best performance when the  $K$  is 40. After increasing the number of topics  $K$ , we can see SAGE consistently increase its accuracy, obtaining its best result when  $K = 30$ . When comparing these two models, SME’s best performance outperforms SAGE’s with an absolute improvement of 3%, which equals to a relative improvement ( $p < 0.001$ ) of 8.4%. Figure 5 demonstrates the impacts of  $K$  on the predictive power of SME and SAGE in the region identification task.

the two tasks in SME have similar difficulties and accuracies, but this needs to be verified in future work.

Keywords discovered by the SME model	
Prior to 1837 (Q1)	pauperis, footprints, American Colonization Society, manumissions, 1797
1838 - 1848 (Q2)	indentured, borrowers, orphan's, 1841, vendee's, drawer's, copartners
1849 - 1855 (Q3)	Frankfort, negro trader, 1851, Kentucky Assembly, marshaled, classed
After 1856 (Q4)	railroadco, statute, Alabama, steamboats, Waterman's, mulattoes, man-trap
Free Region (R1)	apprenticed, overseer's, Federal Army, manumitting, Illinois constitution
Slave Region (R2)	Alabama, Clay's Digest, oldest, cotton, reinstatement, sanction, plantation's
Topic 1 in Q1 R1	imported, comaker, runs, writ's, remainderman's, converters, runaway
Topic 1 in Q1 R2	comaker, imported, deceitful, houston, send, bright, remainderman's
Topic 2 in Q1 R1	descendent, younger, administrator's, documentary, agreeable, emancipated
Topic 2 in Q1 R2	younger, administrator's, grandmother's, plaintiffs, emancipated, learnedly
Topic 3 in Q2 R1	heir-at-law, reconsidered, manumissions, birthplace, mon, mother-in-law
Topic 3 in Q2 R2	heir-at-law, reconsideration, mon, confessions, birthplace, father-in-law's
Topic 4 in Q2 R1	indentured, apprenticed, deputy collector, stepfather's, traded, seizes
Topic 4 in Q2 R2	deputy collector, seizes, traded, hiring, stepfather's, indentured, teaching
Topic 5 in Q4 R1	constitutionality, constitutional, unconstitutionally, Federal Army, violated
Topic 5 in Q4 R2	petition, convictions, criminal court, murdered, constitutionality, man-trap

Table 2: A partial listing of an example for early United States state supreme court opinion keywords generated from the time quartile  $\eta^{(Q)}$ , region  $\eta^{(R)}$  and topic-region-time  $\eta^{(I)}$  interactive variables in the sparse mixed-effects model.

Except that the two models tie up when  $K = 10$ , SME outperforms SAGE for all subsequent variations of  $K$ . Similar to the region task, SME achieves the best result when  $K$  is sparser ( $p < 0.01$  when  $K = 40$  and  $K = 50$ ).

## 5.2 Qualitative Analysis

In this section, we qualitatively evaluate the topics generated vis-a-vis the secondary literature on the legal and political history of slavery in the United States. The effectiveness of SME could depend not just on its predictive power, but also in its ability to generate topics that will be useful to historians of the period. Supreme court opinions on slavery are of significant interest for American political history. The conflict over slave property rights was at the heart of the “cold war” (Wright, 2006) between North and South leading up to the U.S. Civil War. The historical importance of this conflict between Northern and Southern legal institutions is one of the motivations for choosing our data domain.

We conduct qualitative analyses on the top-ranked keywords<sup>6</sup> that are associated with different geographical locations and different temporal frames, generated by our SME model. In our analysis, for

each interaction of topic, region, and time period, a list of the most salient vocabulary words was generated. These words were then analyzed in the context of existing historical literature on the shift in attitudes and views over time and across regions. Table 2 shows an example of relevant keywords and topics.

This difference between Northern and Southern opinion can be seen in some of the topics generated by the SME. Topic 1 deals with transfers of human beings as slave property. The keyword “remainderman” designates a person who inherits or is entitled to inherit property upon the termination of an estate, typically after the death of a property owner, and appears in Northern and Southern cases. However, in Topic 1 “runaway” appears as a keyword in decisions from free states but not in decisions from slave states. The fact that “runaway” is not a top word in the same topic in the Southern legal opinions is consistent with a spatial (geolocational) division in which the property claims of slave owners over runaways were not heavily contested in Southern courts.

Topic 3 concerns bequests, as indicated by the term “heir-at-law”, but again the term “manumissions”, ceases to show up in the slave states after the first time quartile, perhaps reflecting the hostility to

<sup>6</sup>Keywords were ranked by word posterior probabilities.

manumissions that southern courts exhibited as the conflict over slavery deepened.

Topic 4 concerns indentures and apprentices. Interestingly, the terms indentures and apprenticeships are more prominent in the non-slave states, reflecting the fact that apprenticeships and indentures were used in many border states as a substitute for slavery, and these were often governed by continued usage of Master and Servant law (Orren, 1992).

Topic 5 shows the constitutional crisis in the states. In particular, the anti-slavery state courts are prone to use the term “unconstitutional” much more often than the slave states. The word “man-trap”, a term used to refer to states where free blacks could be kidnapped purpose of enslaving them. The fugitive slave conflicts of the mid-19th century that led to the civil war were precisely about this aversion of the northern states to having to return runaway slaves to the Southern states.

Besides these subjective observations about the historical significance of the SME topics, we also conduct a more formal analysis comparing the SME classification to that conducted by a legal historian. Wahl (2002) analyses and classifies by hand 10989 slave cases in the US South into 6 categories: “Hires”, “Sales”, “Transfers”, “Common Carrier”, “Black Rights” and “Other”. An example of “Hires” is Topic 4. Topics 1, 2, and 3 concern “Transfers” of slave property between inheritors, descendants and heirs-at-law. Topic 5 would be classified as “Other”.

We take each of our 25 modelled topics and classify them along Wahl’s categories, using “Other” when a classification could not be obtained. The classifications are quite transparent in virtually all cases, as certain words (such as “employer” or “bequest”) clearly designate certain categories (respectively, such as “Hires” or “Transfers”). We then calculate the probability of each of Wahl’s categories in Region 2. We then compare these to the relative frequencies of Wahl’s categorization in the states that overlap with our Region 2 in Figure 6 and do a  $\chi^2$  test for goodness of fit, which allows us to reject difference at 0.1% confidence.

The SME model thus delivers topics that, at a first pass, are consistent with the history of the period as well as previous work by historians, showing the qualitative benefits of the model. We plan to conduct more vertical and temporal analyses using SME in the future.

## 6 Conclusion and Future Work

In this work, we propose a sparse mixed-effects model for historical analysis of text. This model is built on the state-of-the-art in latent variable mod-

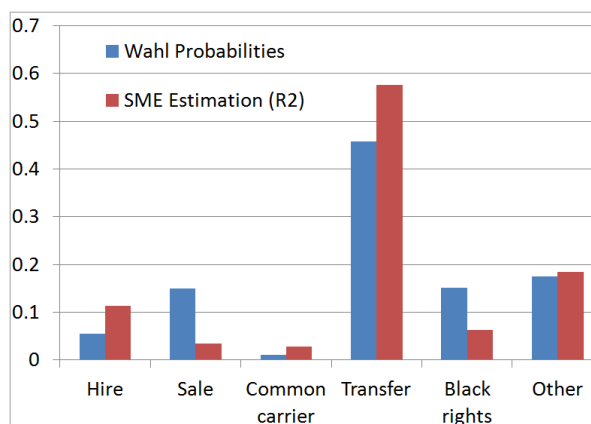


Figure 6: Comparison with Wahl (2002) classification.

elling and extends that model to a setting where metadata is available for analysis. We jointly model those observed labels as well as unsupervised topic modelling. In our experiments, we have shown that the resulting model jointly predicts the region and the time of a given court document. Across vocabulary sizes and number of topics, we have achieved better system accuracy than state-of-the-art generative and discriminative models of text. Our quantitative analysis shows that early US state supreme court opinions are predictable, and contains distinct views towards slave-related topics, and the shifts among opinions depending on different periods of time. In addition, our model has been shown to be effective for qualitative analysis of historical data, revealing patterns that are consistent with the history of the period.

This approach to modelling text is not limited to the legal domain. A key aspect of future work will be to extend the Sparse Mixed-Effects paradigm to other problems within the social sciences where metadata is available but qualitative analysis at a large scale is difficult or impossible. In addition to historical documents, this can include humanities texts, which are often sorely lacking in empirical justifications, and analysis of online communities, which are often rife with available metadata but produce content far faster than it can be analyzed by experts.

## Acknowledgments

We thank Jacob Eisenstein, Noah Smith, and anonymous reviewers for valuable suggestions. William Yang Wang is supported by the R. K. Mellon Presidential Fellowship.



## References

- Lalit R. Bahl, Peter F. Brown., Peter V. de Souza, and Robert L. Mercer. 1988. A new algorithm for the estimation of hidden Markov model parameters. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 493–496.
- Ryan S.J.D. Baker and Kalina Yacef. 2009. The state of educational data mining in 2009: a review and future visions. In *Journal of Educational Data Mining*, pages 3–17.
- David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research (JMLR)*, pages 993–1022.
- Chih-Chung Chang and Chih-Jen Lin. 2011. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent System Technologies*, pages 1–27.
- Jake Chen and Stefano Lombardi. 2010. *Biological data mining*. Chapman and Hall/CRC.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8.
- Agata Katarzyna Cybulska and Piek Vossen. 2011. Historical event extraction from text. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 39–43.
- Jacob Eisenstein, Brendan O’Connor, Noah A. Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 1277–1287.
- Jacob Eisenstein, Amr Ahmed, and Eric. Xing. 2011a. Sparse additive generative models of text. *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, pages 1041–1048.
- Jacob Eisenstein, Noah A. Smith, and Eric P. Xing. 2011b. Discovering sociolinguistic associations with structured sparsity. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, pages 1365–1374.
- Annette Gotscharek, Andreas Neumann, Ulrich Reffle, Christoph Ringlstetter, and Klaus U. Schulz. 2009. Enabling information retrieval on historical document collections: the role of matching procedures and special lexica. In *Proceedings of The Third Workshop on Analytics for Noisy Unstructured Text Data (AND 2009)*, pages 69–76.
- Weiwei Guo and Mona Diab. 2011. Semantic topic models: combining word distributional statistics and dictionary definitions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 552–561.
- Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proceedings of Uncertainty in Artificial Intelligence (UAI 1999)*, pages 289–296.
- Thorsten Joachims. 1998. Text categorization with support vector machines: learning with many relevant features.
- Kenneth Lange and Janet S. Sinsheimer. 1993. Normal/independent distributions and their applications in robust regression.
- Dong Nguyen and Carolyn Penstein Rosé. 2011. Language use as a reflection of socialization in online communities. In *Workshop on Language in Social Media at ACL*.
- Dong Nguyen, Elijah Mayfield, and Carolyn P. Rosé. 2010. An analysis of perspectives in interactive settings. In *Proceedings of the First Workshop on Social Media Analytics (SOMA 2010)*, pages 44–52.
- Karen Orren. 1992. Belated feudalism: labor, the law, and liberal development in the united states.
- Eva Pettersson and Joakim Nivre. 2011. Automatic verb extraction from historical swedish texts. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 87–95.
- William D. Popkin. 2007. *Evolution of the judicial opinion: institutional and individual styles*. NYU Press.
- Kumar Shubhankar, Aditya Pratap Singh, and Vikram Pudi. 2011. An efficient algorithm for topic ranking and modeling topic evolution. In *Proceedings of International Conference on Database and Expert Systems Applications*.
- Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, pages 1566–1581.
- Jenny Bourne Wahl. 2002. *The Bondsman’s Burden: An Economic Analysis of the Common Law of Southern Slavery*. Cambridge University Press.
- William Yang Wang and Kathleen McKeown. 2010. ”got you!”: automatic vandalism detection in wikipedia with web-based shallow syntactic-semantic modeling. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1146–1154.
- William Yang Wang, Kapil Thadani, and Kathleen McKeown. 2011. Identifying event descriptions using co-training with online news summaries. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 281–291.



- Gavin Wright. 2006. Slavery and american economic development. *Walter Lynwood Fleming Lectures in Southern History*.
- Tze-I Yang, Andrew Torget, and Rada Mihalcea. 2011. Topic modeling on historical newspapers. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 96–104.

# A Topic Similarity Model for Hierarchical Phrase-based Translation

Xinyan Xiao<sup>†</sup> Deyi Xiong<sup>‡</sup> Min Zhang<sup>‡\*</sup> Qun Liu<sup>†</sup> Shouxun Lin<sup>†</sup>

<sup>†</sup>Key Lab. of Intelligent Info. Processing  
Institute of Computing Technology  
Chinese Academy of Sciences

<sup>‡</sup>Human Language Technology  
Institute for Infocomm Research

{xiaoxinyan, liuqun, sxlin}@ict.ac.cn {dyxiong, mzhang\*}@i2r.a-star.edu.sg

## Abstract

Previous work using topic model for statistical machine translation (SMT) explore topic information at the word level. However, SMT has been advanced from word-based paradigm to phrase/rule-based paradigm. We therefore propose a topic similarity model to exploit topic information at the synchronous rule level for hierarchical phrase-based translation. We associate each synchronous rule with a topic distribution, and select desirable rules according to the similarity of their topic distributions with given documents. We show that our model significantly improves the translation performance over the baseline on NIST Chinese-to-English translation experiments. Our model also achieves a better performance and a faster speed than previous approaches that work at the word level.

## 1 Introduction

Topic model (Hofmann, 1999; Blei et al., 2003) is a popular technique for discovering the underlying topic structure of documents. To exploit topic information for statistical machine translation (SMT), researchers have proposed various topic-specific lexicon translation models (Zhao and Xing, 2006; Zhao and Xing, 2007; Tam et al., 2007) to improve translation quality.

Topic-specific lexicon translation models focus on word-level translations. Such models first estimate word translation probabilities conditioned on topics, and then adapt lexical weights of phrases

by these probabilities. However, the state-of-the-art SMT systems translate sentences by using sequences of synchronous rules or phrases, instead of translating word by word. Since a synchronous rule is rarely factorized into individual words, we believe that it is more reasonable to incorporate the topic model directly at the rule level rather than the word level.

Consequently, we propose a **topic similarity** model for hierarchical phrase-based translation (Chiang, 2007), where each synchronous rule is associated with a topic distribution. In particular,

- Given a document to be translated, we calculate the topic similarity between a rule and the document based on their topic distributions. We augment the hierarchical phrase-based system by integrating the proposed topic similarity model as a new feature (Section 3.1).
- As we will discuss in Section 3.2, the similarity between a generic rule and a given source document computed by our topic similarity model is often very low. We don't want to penalize these generic rules. Therefore we further propose a topic sensitivity model which rewards generic rules so as to complement the topic similarity model.
- We estimate the topic distribution for a rule based on both the source and target side topic models (Section 4.1). In order to calculate similarities between target-side topic distributions of rules and source-side topic distributions of given documents during decoding, we project

\*Corresponding author

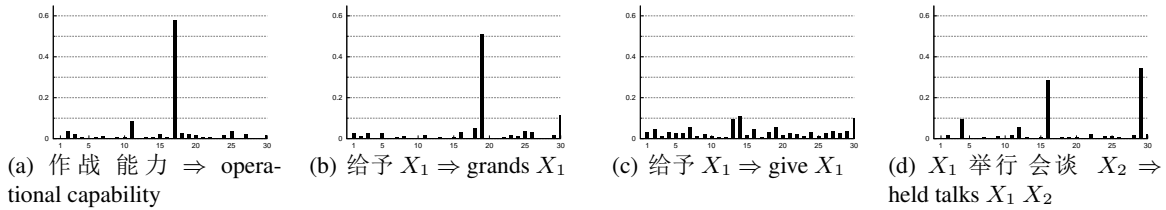


Figure 1: Four synchronous rules with topic distributions. Each sub-graph shows a rule with its topic distribution, where the X-axis means topic index and the Y-axis means the topic probability. Notably, the rule (b) and rule (c) shares the same source Chinese string, but they have different topic distributions due to the different English translations.

the target-side topic distributions of rules into the space of source-side topic model by one-to-many projection (Section 4.2).

Experiments on Chinese-English translation tasks (Section 6) show that, our method outperforms the baseline hierarchical phrase-based system by +0.9 BLEU points. This result is also +0.5 points higher and 3 times faster than the previous topic-specific lexicon translation method. We further show that both the source-side and target-side topic distributions improve translation quality and their improvements are complementary to each other.

## 2 Background: Topic Model

A topic model is used for discovering the topics that occur in a collection of documents. Both Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and Probabilistic Latent Semantic Analysis (PLSA) (Hofmann, 1999) are types of topic models. LDA is the most common topic model currently in use, therefore we exploit it for mining topics in this paper. Here, we first give a brief description of LDA.

LDA views each document as a mixture proportion of various topics, and generates each word by multinomial distribution conditioned on a topic. More specifically, as a generative process, LDA first samples a document-topic distribution for each document. Then, for each word in the document, it samples a topic index from the document-topic distribution and samples the word conditioned on the topic index according the topic-word distribution.

Generally speaking, LDA contains two types of parameters. The first one relates to the document-topic distribution, which records the topic distribution of each document. The second one is used for topic-word distribution, which represents each topic

as a distribution over words. Based on these parameters (and some hyper-parameters), LDA can infer a topic assignment for each word in the documents. In the following sections, we will use these parameters and the topic assignments of words to estimate the parameters in our method.

## 3 Topic Similarity Model

Sentences should be translated in consistence with their topics (Zhao and Xing, 2006; Zhao and Xing, 2007; Tam et al., 2007). In the hierarchical phrase based system, a synchronous rule may be related to some topics and unrelated to others. In terms of probability, a rule often has an uneven probability distribution over topics. The probability over a topic is high if the rule is highly related to the topic, otherwise the probability will be low. Therefore, we use topic distribution to describe the relatedness of rules to topics.

Figure 1 shows four synchronous rules (Chiang, 2007) with topic distributions, some of which contain nonterminals. We can see that, although the source part of rule (b) and (c) are identical, their topic distributions are quite different. Rule (b) contains a highest probability on the topic about “China-U.S. relationship”, which means rule (b) is much more related to this topic. In contrast, rule (c) contains an even distribution over various topics. Thus, given a document about “China-U.S. relationship”, we hope to encourage the system to apply rule (b) but penalize the application of rule (c). We achieve this by calculating similarity between the topic distributions of a rule and a document to be translated.

More formally, we associate each rule with a **rule-topic distribution**  $P(z|r)$ , where  $r$  is a rule, and  $z$  is a topic. Suppose there are  $K$  topics, this distribution

can be represented by a  $K$ -dimension vector. The  $k$ -th component  $P(z = k|r)$  means the probability of topic  $k$  given the rule  $r$ . The estimation of such distribution will be described in Section 4.

Analogously, we represent the topic information of a document  $d$  to be translated by a **document-topic distribution**  $P(z|d)$ , which is also a  $K$ -dimension vector. The  $k$ -th dimension  $P(z = k|d)$  means the probability of topic  $k$  given document  $d$ . Different from rule-topic distribution, the document-topic distribution can be directly inferred by an off-the-shelf LDA tool.

Consequently, based on these two distributions, we select a rule for a document to be translated according to their **topic similarity** (Section 3.1), which measures the relatedness of the rule to the document. In order to encourage the application of generic rules which are often penalized by our similarity model, we also propose a topic sensitivity model (Section 3.2).

### 3.1 Topic Similarity

By comparing the similarity of their topic distributions, we are able to decide whether a rule is suitable for a given source document. The topic similarity computes the distance of two topic distributions. We calculate the topic similarity by Hellinger function:

$$\begin{aligned} & \text{Similarity}(P(z|d), P(z|r)) \\ &= \sum_{k=1}^K \left( \sqrt{P(z = k|d)} - \sqrt{P(z = k|r)} \right)^2 \quad (1) \end{aligned}$$

Hellinger function is used to calculate distribution distance and is popular in topic model (Blei and Lafferty, 2007).<sup>1</sup> By topic similarity, we aim to encourage or penalize the application of a rule for a given document according to their topic distributions, which then helps the SMT system make better translation decisions.

### 3.2 Topic Sensitivity

Domain adaptation (Wu et al., 2008; Bertoldi and Federico, 2009) often distinguishes general-domain data from in-domain data. Similarly, we divide the rules into topic-insensitive rules and topic-sensitive

<sup>1</sup>We also try other distance functions, including Euclidean distance, Kullback-Leibler divergence and cosine function. They produce similar results in our preliminary experiments.

rules according to their topic distributions. Let's revisit Figure 1. We can easily find that the topic distribution of rule (c) distribute evenly. This indicates that it is insensitive to topics, and can be applied in any topics. We call such a rule a topic-insensitive rule. In contrast, the distributions of the rest rules peak on a few topics. Such rules are called topic-sensitive rules. Generally speaking, a topic-insensitive rule has a fairly flat distribution, while a topic-sensitive rule has a sharp distribution.

A document typically focuses on a few topics, and has a sharp topic distribution. In contrast, the distribution of topic-insensitive rule is fairly flat. Hence, a topic-insensitive rule is always less similar to documents and is punished by the similarity function.

However, topic-insensitive rules may be more preferable than topic-sensitive rules if neither of them are similar to given documents. For a document about the "military" topic, the rule (b) and (c) in Figure 1 are both dissimilar to the document, because rule (b) relates to the "China-U.S. relationship" topic and rule (c) is topic-insensitive. Nevertheless, since rule (c) occurs more frequently across various topics, it may be better to apply rule (c).

To address such issue of the topic similarity model, we further introduce a topic sensitivity model to describe the topic sensitivity of a rule using entropy as a metric:

$$\begin{aligned} & \text{Sensitivity}(P(z|r)) \\ &= - \sum_{k=1}^K P(z = k|r) \times \log(P(z = k|r)) \quad (2) \end{aligned}$$

According to the Eq. (2), a topic-insensitive rule has a large entropy, while a topic-sensitive rule has a smaller entropy. By incorporating the topic sensitivity model with the topic similarity model, we enable our SMT system to balance the selection of these two types of rules. Given rules with approximately equal values of Eq. (1), we prefer topic-insensitive rules.

## 4 Estimation

Unlike document-topic distribution that can be directly learned by LDA tools, we need to estimate the rule-topic distribution according to our requirement. In this paper, we try to exploit the topic information

of both source and target language. To achieve this goal, we use both source-side and target-side monolingual topic models, and learn the correspondence between the two topic models from word-aligned bilingual corpus.

Specifically, we use two types of rule-topic distributions: one is source-side rule-topic distribution and the other is target-side rule-topic distribution. These two rule-topic distributions are estimated by corresponding topic models in the same way (Section 4.1). Notably, only source language documents are available during decoding. In order to compute the similarity between the target-side topic distribution of a rule and the source-side topic distribution of a given document, we need to project the target-side topic distribution of a synchronous rule into the space of the source-side topic model (Section 4.2).

A more principle way is to learn a bilingual topic model from bilingual corpus (Mimno et al., 2009). However, we may face difficulty during decoding, where only source language documents are available. It requires a marginalization to infer the monolingual topic distribution using the bilingual topic model. The high complexity of marginalization prohibits such a summation in practice. Previous work on bilingual topic model avoid this problem by some monolingual assumptions. Zhao and Xing (2007) assume that the topic model is generated in a monolingual manner, while Tam et al., (2007) construct their bilingual topic model by enforcing a one-to-one correspondence between two monolingual topic models. We also estimate our rule-topic distribution by two monolingual topic models, but use a different way to project target-side topics onto source-side topics.

#### 4.1 Monolingual Topic Distribution Estimation

We estimate rule-topic distribution from word-aligned bilingual training corpus with document boundaries explicitly given. The source and target side distributions are estimated in the same way. For simplicity, we only describe the estimation of source-side distribution in this section.

The process of rule-topic distribution estimation is analogous to the traditional estimation of rule translation probability (Chiang, 2007). In addition to the word-aligned corpus, the input for estimation also contains the source-side topic-document distri-

bution of every documents inferred by LDA tool.

We first extract synchronous rules from training data in a traditional way. When a rule  $r$  is extracted from a document  $d$  with topic distribution  $P(z|d)$ , we collect an instance  $(r, P(z|d), c)$ , where  $c$  is the fraction count of an instance as described in Chiang, (2007). After extraction, we get a set of instances  $\mathcal{I} = \{(r, P(z|d), c)\}$  with different document-topic distributions for each rule. Using these instances, we calculate the topic probability  $P(z = k|r)$  as follows:

$$P(z = k|r) = \frac{\sum_{I \in \mathcal{I}} c \times P(z = k|d)}{\sum_{k'=1}^K \sum_{I \in \mathcal{I}} c \times P(z = k'|d)} \quad (3)$$

By using both source-side and target-side document-topic distribution, we obtain two rule-topic distributions for each rule in total.

#### 4.2 Target-side Topic Distribution Projection

As described in the previous section, we also estimate the target-side rule-topic distribution. However, only source document-topic distributions are available during decoding. In order to calculate the similarity between the target-side rule-topic distribution of a rule and the source-side document-topic distribution of a source document, we need to project target-side topics into the source-side topic space. The projection contains two steps:

- In the first step, we learn the topic-to-topic correspondence probability  $p(z_f|z_e)$  from target-side topic  $z_e$  to source-side topic  $z_f$ .
- In the second step, we project the target-side topic distribution of a rule into source-side topic space using the correspondence probability.

In the first step, we estimate the correspondence probability by the co-occurrence of the source-side and the target-side topic assignment of the word-aligned corpus. The topic assignments are output by LDA tool. Thus, we denotes each sentence pair by  $(\mathbf{z}_f, \mathbf{z}_e, \mathbf{a})$ , where  $\mathbf{z}_f$  and  $\mathbf{z}_e$  are the topic assignments of source-side and target-side sentences respectively, and  $\mathbf{a}$  is a set of links  $\{(i, j)\}$ . A link  $(i, j)$  means a source-side position  $i$  aligns to a target-side position  $j$ . Thus, the co-occurrence of a source-side topic with index  $k_f$  and a target-side

e-topic	f-topic 1	f-topic 2	f-topic 3
enterprises	农业(agricultural)	企业(enterprise)	发展(develop)
rural	农村(rural)	市场(market)	经济(economic)
state	农民(peasant)	国有(state)	科技(technology)
agricultural	改革(reform)	公司(company)	我国(China)
market	财政(finance)	金融(finance)	技术(technique)
reform	社会(social)	银行(bank)	产业(industry)
production	保障(safety)	投资(investment)	结构(structure)
peasants	调整(adjust)	管理(manage)	创新(innovation)
owned	政策(policy)	改革(reform)	加快(accelerate)
enterprise	收入(income)	经营(operation)	改革(reform)
$p(z_f z_e)$	0.38	0.28	0.16

Table 1: Example of topic-to-topic correspondence. The last line shows the correspondence probability. Each column means a topic represented by its top-10 topical words. The first column is a target-side topic, while the rest three columns are source-side topics.

topic  $k_e$  is calculated by:

$$\sum_{(\mathbf{z}_f, \mathbf{z}_e, \mathbf{a})} \sum_{(i, j) \in \mathbf{a}} \delta(z_{f_i}, k_f) * \delta(z_{e_j}, k_e) \quad (4)$$

where  $\delta(x, y)$  is the Kronecker function, which is 1 if  $x = y$  and 0 otherwise. We then compute the probability of  $P(z = k_f | z = k_e)$  by normalizing the co-occurrence count. Overall, after the first step, we obtain an correspondence matrix  $\mathbf{M}_{K_e \times K_f}$  from target-side topic to source-side topic, where the item  $M_{i,j}$  represents the probability  $P(z_f = i | z_e = j)$ .

In the second step, given the correspondence matrix  $\mathbf{M}_{K_e \times K_f}$ , we project the target-side rule-topic distribution  $P(z_e|r)$  to the source-side topic space by multiplication as follows:

$$T(P(z_e|r)) = P(z_e|r) \otimes \mathbf{M}_{K_e \times K_f} \quad (5)$$

In this way, we get a second distribution for a rule in the source-side topic space, which we called projected target-side topic distribution  $T(P(z_e|r))$ .

Obviously, our projection method allows one target-side topic to align to multiple source-side topics. This is different from the one-to-one correspondence used by Tam et al., (2007). From the training result of the correspondence matrix  $\mathbf{M}_{K_e \times K_f}$ , we find that the topic correspondence between source and target language is not necessarily one-to-one. Typically, the probability  $P(z = k_f | z = k_e)$  of a target-side topic mainly distributes on two or three source-side topics. Table 1 shows an example of a target-side topic with its three mainly aligned source-side topics.

## 5 Decoding

We incorporate our topic similarity model as a new feature into a traditional hiero system (Chiang, 2007) under discriminative framework (Och and Ney, 2002). Considering there are a source-side rule-topic distribution and a projected target-side rule-topic distribution, we add four features in total:

- *Similarity* ( $P(z_f|d), P(z_f|r)$ )
- *Similarity* ( $P(z_f|d), T(P(z_e|r))$ )
- *Sensitivity* ( $P(z_f|r)$ )
- *Sensitivity* ( $T(P(z_e|r))$ )

To calculate the total score of a derivation on each feature listed above during decoding, we sum up the correspondent feature score of each applied rule.<sup>2</sup>

The source-side and projected target-side rule-topic distribution are calculated before decoding. During decoding, we first infer the topic distribution  $P(z_f|d)$  for a given document on source language. When applying a rule, it is straightforward to calculate these topic features. Obviously, the computational cost of these features is rather small.

In the topic-specific lexicon translation model, given a source document, it first calculates the topic-specific translation probability by normalizing the entire lexicon translation table, and then adapts the lexical weights of rules correspondingly. This makes the decoding slower. Therefore, comparing with the previous topic-specific lexicon translation method, our method provides a more efficient way for incorporating topic model into SMT.

## 6 Experiments

We try to answer the following questions by experiments:

1. Is our topic similarity model able to improve translation quality in terms of BLEU? Furthermore, are source-side and target-side rule-topic distributions complementary to each other?

<sup>2</sup>Since glue rule and rules of unknown words are not extracted from training data, here, we just ignore the calculation of the four features for them.

System	MT06	MT08	Avg	Speed
Baseline	30.20	21.93	26.07	12.6
TopicLex	30.65	22.29	26.47	3.3
SimSrc	30.41	<b>22.69</b>	26.55	11.5
SimTgt	30.51	22.39	26.45	11.7
SimSrc+SimTgt	30.73	<b>22.69</b>	<b>26.71</b>	11.2
Sim+Sen	<b>30.95</b>	<b>22.92</b>	<b>26.94</b>	10.2

Table 2: Result of our topic similarity model in terms of BLEU and speed (words per second), comparing with the traditional hierarchical system (“Baseline”) and the topic-specific lexicon translation method (“TopicLex”). “SimSrc” and “SimTgt” denote similarity by source-side and target-side rule-distribution respectively, while “Sim+Sen” activates the two similarity and two sensitivity features. “Avg” is the average BLEU score on the two test sets. Scores marked in bold mean significantly (Koehn, 2004) better than *Baseline* ( $p < 0.01$ ).

2. Is it helpful to introduce the topic sensitivity model to distinguish topic-insensitive and topic-sensitive rules?
3. Is it necessary to project topics by one-to-many correspondence instead of one-to-one correspondence?
4. What is the effect of our method on various types of rules, such as phrase rules and rules with non-terminals?

## 6.1 Data

We present our experiments on the NIST Chinese-English translation tasks. The bilingual training data contains 239K sentence pairs with 6.9M Chinese words and 9.14M English words, which comes from the FBIS portion of LDC data. There are 10,947 documents in the FBIS corpus. The monolingual data for training English language model includes the Xinhua portion of the GIGAWORD corpus, which contains 238M English words. We used the NIST evaluation set of 2005 (MT05) as our development set, and sets of MT06/MT08 as test sets. The numbers of documents in MT05, MT06, MT08 are 100, 79, and 109 respectively.

We obtained symmetric word alignments of training data by first running GIZA++ (Och and Ney, 2003) in both directions and then applying refinement rule “grow-diag-final-and” (Koehn et al., 2003). The SCFG rules are extracted from this word-aligned training data. A 4-gram language model was trained on the monolingual data by the SRILM toolkit (Stolcke, 2002). Case-insensitive NIST BLEU (Papineni et al., 2002) was used to mea-

sure translation performance. We used minimum error rate training (Och, 2003) for optimizing the feature weights.

For the topic model, we used the open source LDA tool GibbsLDA++ for estimation and inference.<sup>3</sup> GibbsLDA++ is an implementation of LDA using gibbs sampling for parameter estimation and inference. The source-side and target-side topic models are estimated from the Chinese part and English part of FBIS corpus respectively. We set the number of topic  $K = 30$  for both source-side and target-side, and use the default setting of the tool for training and inference.<sup>4</sup> During decoding, we first infer the topic distribution of given documents before translation according to the topic model trained on Chinese part of FBIS corpus.

## 6.2 Effect of Topic Similarity Model

We compare our method with two baselines. In addition to the traditional hiero system, we also compare with the topic-specific lexicon translation method in Zhao and Xing (2007). The lexicon translation probability is adapted by:

$$\begin{aligned}
 p(f|e, D_F) &\propto p(e|f, D_F)P(f|D_F) & (6) \\
 &= \sum_k p(e|f, z = k)p(f|z = k)p(z = k|D_F) & (7)
 \end{aligned}$$

However, we simplify the estimation of  $p(e|f, z = k)$  by directly using the word alignment corpus with

<sup>3</sup><http://gibbslda.sourceforge.net/>

<sup>4</sup>We determine  $K$  by testing {15, 30, 50, 100, 200} in our preliminary experiments. We find that  $K = 30$  produces a slightly better performance than other values.

Type	Count	Src%	Tgt%
Phrase-rule	3.9M	83.4	84.4
Monotone-rule	19.2M	85.3	86.1
Reordering-rule	5.7M	85.9	86.8
All-rule	28.8M	85.1	86.0

Table 3: Percentage of topic-sensitive rules of various types of rule according to source-side (“Src”) and target-side (“Tgt”) topic distributions. Phrase rules are fully lexicalized, while monotone and reordering rules contain nonterminals (Section 6.5).

topic assignment that is inferred by the GibbsL-DA++. Despite the simplification of estimation, the improvement of our implementation is comparable with the improvement in Zhao et al.,(2007). Given a new document, we need to adapt the lexical translation weights of the rules based on topic model. The adapted lexicon translation model is added as a new feature under the discriminative framework.

Table 2 shows the result of our method comparing with the traditional system and the topic-lexicon specific translation method described as above. By using all the features (last line in the table), we improve the translation performance over the baseline system by 0.87 BLEU point on average. Our method also outperforms the topic-lexicon specific translation method by 0.47 points. This verifies that topic similarity model can improve the translation quality significantly.

In order to gain insights into why our model is helpful, we further investigate how many rules are topic-sensitive. As described in Section 3.2, we use entropy to measure the topic sensitivity. If the entropy of a rule is smaller than a certain threshold, then the rule is topic sensitive. Since documents often focus on some topics, we use the average entropy of document-topic distribution of all training documents as the threshold. We compare both source-side and target-side distribution shown in Table 3. We find that more than 80 percents of the rules are topic-sensitive, thus provides us a large space to improve the translation by exploiting topics.

We also compare these methods in terms of the decoding speed (words/second). The baseline translates 12.6 words per second, while the topic-specific lexicon translation method only translates 3.3 words in one second. The overhead of the topic-specific

System	MT06	MT08	Avg
Baseline	30.20	21.93	26.07
One-to-One	30.27	22.12	26.20
One-to-Many	30.51	22.39	26.45

Table 4: Effects of one-to-one and one-to-many topic projection.

lexicon translation method mainly comes from the adaptation of lexical weights. It takes 72.8% of the time to do the adaptation, despite only lexical weights of the used rules are adapted. In contrast, our method has a speed of 10.2 words per second for each sentence on average, which is three times faster than the topic-specific lexicon translation method.

Meanwhile, we try to separate the effects of source-side topic distribution from the target-side topic distribution. From lines 4-6 of Table 2. We clearly find that the two rule-topic distributions improve the performance by 0.48 and 0.38 BLEU points over the baseline respectively. It seems that the source-side topic model is more helpful. Furthermore, when combine these two distributions, the improvement is increased to 0.64 points. This indicates that the effects of source-side and target-side distributions are complementary.

### 6.3 Effect of Topic Sensitivity Model

As described in Section 3.2, because the similarity features always punish topic-insensitive rules, we introduce topic sensitivity features as a complement. In the last line of Table 2, we obtain a further improvement of 0.23 points, when incorporating topic sensitivity features with topic similarity features. This suggests that it is necessary to distinguish topic-insensitive and topic-sensitive rules.

### 6.4 One-to-One Vs. One-to-Many Topic Projection

In Section 4.2, we find that source-side topic and target-side topics may not exactly match, hence we use one-to-many topic correspondence. Yet another method is to enforce one-to-one topic projection (Tam et al., 2007). We achieve one-to-one projection by aligning a target topic to the source topic with the largest correspondence probability as calculated in Section 4.2.

Table 4 compares the effects of these two method-



System	MT06	MT08	Avg
Baseline	30.20	21.93	26.07
Phrase-rule	30.53	22.29	26.41
Monotone-rule	30.72	22.62	26.67
Reordering-rule	30.31	22.40	26.36
All-rule	30.95	22.92	26.94

Table 5: Effect of our topic model on three types of rules. Phrase rules are fully lexicalized, while monotone and reordering rules contain nonterminals.

s. We find that the enforced one-to-one topic method obtains a slight improvement over the baseline system, while one-to-many projection achieves a larger improvement. This confirms our observation of the non-one-to-one mapping between source-side and target-side topics.

### 6.5 Effect on Various Types of Rules

To get a more detailed analysis of the result, we further compare the effect of our method on different types of rules. We divide the rules into three types: phrase rules, which only contain terminals and are the same as the phrase pairs in phrase-based system; monotone rules, which contain non-terminals and produce monotone translations; reordering rules, which also contain non-terminals but change the order of translations. We define the monotone and reordering rules according to Chiang et al., (2008).

Table 5 show the results. We can see that our method achieves improvements on all the three types of rules. Our topic similarity method on monotone rule achieves the most improvement which is 0.6 BLEU points, while the improvement on reordering rules is the smallest among the three types. This shows that topic information also helps the selections of rules with non-terminals.

## 7 Related Work

In addition to the topic-specific lexicon translation method mentioned in the previous sections, researchers also explore topic model for machine translation in other ways.

Foster and Kunh (2007) describe a mixture-model approach for SMT adaptation. They first split a training corpus into different domains. Then, they train separate models on each domain. Finally, they

combine a specific domain translation model with a general domain translation model depending on various text distances. One way to calculate the distance is using topic model.

Gong et al. (2010) introduce topic model for filtering topic-mismatched phrase pairs. They first assign a specific topic for the document to be translated. Similarly, each phrase pair is also assigned with one specific topic. A phrase pair will be discarded if its topic mismatches the document topic.

Researchers also introduce topic model for cross-lingual language model adaptation (Tam et al., 2007; Ruiz and Federico, 2011). They use bilingual topic model to project latent topic distribution across languages. Based on the bilingual topic model, they apply the source-side topic weights into the target-side topic model, and adapt the n-gram language model of target side.

Our topic similarity model uses the document topic information. From this point, our work is related to context-dependent translation (Carpuat and Wu, 2007; He et al., 2008; Shen et al., 2009). Previous work typically use neighboring words and sentence level information, while our work extents the context into the document level.

## 8 Conclusion and Future Work

We have presented a topic similarity model which incorporates the rule-topic distributions on both the source and target side into traditional hierarchical phrase-based system. Our experimental results show that our model achieves a better performance with faster decoding speed than previous work on topic-specific lexicon translation. This verifies the advantage of exploiting topic model at the rule level over the word level. Further improvement is achieved by distinguishing topic-sensitive and topic-insensitive rules using the topic sensitivity model.

In the future, we are interesting to find ways to exploit topic model on bilingual data without document boundaries, thus to enlarge the size of training data. Furthermore, our training corpus mainly focus on news, it is also interesting to apply our method on corpus with more diverse topics. Finally, we hope to apply our method to other translation models, especially syntax-based models.

## Acknowledgement

The authors were supported by High-Technology R&D Program (863) Project No 2011AA01A207 and 2012BAH39B03. This work was done during Xinyan Xiao's internship at I<sup>2</sup>R. We would like to thank Yun Huang, Zhengxian Gong, Wenliang Chen, Jun lang, Xiangyu Duan, Jun Sun, Jinsong Su and the anonymous reviewers for their insightful comments.

## References

- Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proc of WMT 2009*.
- David M. Blei and John D. Lafferty. 2007. A correlated topic model of science. *AAS*, 1(1):17–35.
- David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent dirichlet allocation. *JMLR*, 3:993–1022.
- Marine Carpuat and Dekai Wu. 2007. Context-dependent phrasal translation lexicons for statistical machine translation. In *Proceedings of the MT Summit XI*.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proc. EMNLP 2008*.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- George Foster and Roland Kuhn. 2007. Mixture-model adaptation for SMT. In *Proc. of the Second Workshop on Statistical Machine Translation*, pages 128–135, Prague, Czech Republic, June.
- Zhengxian Gong, Yu Zhang, and Guodong Zhou. 2010. Statistical machine translation based on lda. In *Proc. IUCS 2010*, page 286 – 290, Oct.
- Zhongjun He, Qun Liu, and Shouxun Lin. 2008. Improving statistical machine translation using lexicalized rule selection. In *Proc. EMNLP 2008*.
- Thomas Hofmann. 1999. Probabilistic latent semantic analysis. In *Proc. of UAI 1999*, pages 289–296.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT-NAACL 2003*.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP 2004*.
- David Mimno, Hanna M. Wallach, Jason Naradowsky, David A. Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Proc. of EMNLP 2009*.
- Franz J. Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. ACL 2002*.
- Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.
- Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL 2003*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. ACL 2002*.
- Nick Ruiz and Marcello Federico. 2011. Topic adaptation for lecture translation through bilingual latent semantic models. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, July.
- Libin Shen, Jinxi Xu, Bing Zhang, Spyros Matsoukas, and Ralph Weischedel. 2009. Effective use of linguistic and contextual information for statistical machine translation. In *Proc. EMNLP 2009*.
- Andreas Stolcke. 2002. Srilm – an extensible language modeling toolkit. In *Proc. ICSLP 2002*.
- Yik-Cheung Tam, Ian R. Lane, and Tanja Schultz. 2007. Bilingual lsa-based adaptation for statistical machine translation. *Machine Translation*, 21(4):187–207.
- Hua Wu, Haifeng Wang, and Chengqing Zong. 2008. Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proc. Coling 2008*.
- Bing Zhao and Eric P. Xing. 2006. BiTAM: Bilingual topic admixture models for word alignment. In *Proc. ACL 2006*.
- Bin Zhao and Eric P. Xing. 2007. HM-BiTAM: Bilingual topic exploration, word alignment, and translation. In *Proc. NIPS 2007*.

# Modeling Topic Dependencies in Hierarchical Text Categorization

**Alessandro Moschitti and Qi Ju**

University of Trento  
38123 Povo (TN), Italy  
{moschitti, qi}@disi.unitn.it

**Richard Johansson**

University of Gothenburg  
SE-405 30 Gothenburg, Sweden  
richard.johansson@gu.se

## Abstract

In this paper, we encode topic dependencies in hierarchical multi-label Text Categorization (TC) by means of rerankers. We represent reranking hypotheses with several innovative kernels considering both the structure of the hierarchy and the probability of nodes. Additionally, to better investigate the role of category relationships, we consider two interesting cases: (i) traditional schemes in which node-fathers include all the documents of their child-categories; and (ii) more general schemes, in which children can include documents not belonging to their fathers. The extensive experimentation on Reuters Corpus Volume 1 shows that our rerankers inject effective structural semantic dependencies in multi-classifiers and significantly outperform the state-of-the-art.

## 1 Introduction

Automated Text Categorization (TC) algorithms for hierarchical taxonomies are typically based on flat schemes, e.g., one-vs.-all, which do not take topic relationships into account. This is due to two major problems: (i) complexity in introducing them in the learning algorithm and (ii) the small or no advantage that they seem to provide (Rifkin and Klautau, 2004).

We speculate that the failure of using hierarchical approaches is caused by the inherent complexity of modeling all possible topic dependencies rather than the uselessness of such relationships. More precisely, although hierarchical multi-label classifiers can exploit machine learning algorithms for structural output, e.g., (Tsochantaridis et al., 2005; Riezler and Vasserman, 2010; Lavergne et al., 2010),

they often impose a number of simplifying restrictions on some category assignments. Typically, the probability of a document  $d$  to belong to a subcategory  $C_i$  of a category  $C$  is assumed to depend only on  $d$  and  $C$ , but not on other subcategories of  $C$ , or any other categories in the hierarchy. Indeed, the introduction of these long-range dependencies lead to computational intractability or more in general to the problem of how to select an effective subset of them. It is important to stress that (i) there is no theory that can suggest which are the dependencies to be included in the model and (ii) their exhaustive explicit generation (i.e., the generation of all hierarchy subparts) is computationally infeasible. In this perspective, kernel methods are a viable approach to implicitly and easily explore feature spaces encoding dependencies. Unfortunately, structural kernels, e.g., tree kernels, cannot be applied in structured output algorithms such as (Tsochantaridis et al., 2005), again for the lack of a suitable theory.

In this paper, we propose to use the combination of *reranking* with kernel methods as a way to handle the computational and feature design issues. We first use a basic hierarchical classifier to generate a hypothesis set of limited size, and then apply reranking models. Since our rerankers are simple binary classifiers of hypothesis pairs, they can encode complex dependencies thanks to kernel methods. In particular, we used tree, sequence and linear kernels applied to structural and feature-vector representations describing hierarchical dependencies.

Additionally, to better investigate the role of topical relationships, we consider two interesting cases: (i) traditional categorization schemes in which node-

fathers include all the documents of their child-categories; and (ii) more general schemes, in which children can include documents not belonging to their fathers. The intuition under the above setting is that shared documents between categories create semantic links between them. Thus, if we remove common documents between father and children, we reduce the dependencies that can be captured with traditional bag-of-words representation.

We carried out experiments on two entire hierarchies TOPICS (103 nodes organized in 5 levels) and INDUSTRIAL (365 nodes organized in 6 levels) of the well-known Reuters Corpus Volume 1 (RCV1). We first evaluate the accuracy as well as the efficiency of several reranking models. The results show that all our rerankers consistently and significantly improve on the traditional approaches to TC up to 10 absolute percent points. Very interestingly, the combination of structural kernels with the linear kernel applied to vectors of category probabilities further improves on reranking: such a vector provides a more effective information than the joint global probability of the reranking hypothesis.

In the rest of the paper, Section 2 describes the hypothesis generation algorithm, Section 3 illustrates our reranking approach based on tree kernels, Section 4 reports on our experiments, Section 5 illustrates the related work and finally Section 6 derives the conclusions.

## 2 Hierarchy classification hypotheses from binary decisions

The idea of the paper is to build efficient models for hierarchical classification using global dependencies. For this purpose, we use reranking models, which encode global information. This necessitates of a set of initial hypotheses, which are typically generated by *local classifiers*. In our study, we used  $n$  one-vs.-all binary classifiers, associated with the  $n$  different nodes of the hierarchy. In the following sections, we describe a simple framework for hypothesis generation.

### 2.1 Top $k$ hypothesis generation

Given  $n$  categories,  $C_1, \dots, C_n$ , we can define  $p_{C_i}^1(d)$  and  $p_{C_i}^0(d)$  as the probabilities that the classifier  $i$  assigns the document  $d$  to  $C_i$  or not, respectively. For example,  $p_{C_i}^h(d)$  can be computed from

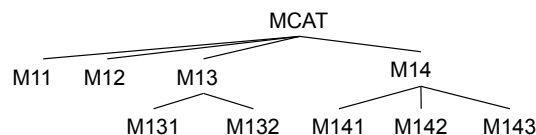


Figure 1: A subhierarchy of Reuters.



Figure 2: A tree representing a category assignment hypothesis for the subhierarchy in Fig. 1.

the SVM classification output (i.e., the example margin). Typically, a large margin corresponds to high probability for  $d$  to be in the category whereas small margin indicates low probability<sup>1</sup>. Let us indicate with  $\mathbf{h} = \{h_1, \dots, h_n\} \in \{0, 1\}^n$  a classification hypothesis, i.e., the set of  $n$  binary decisions for a document  $d$ . If we assume independence between the SVM scores, the most probable hypothesis on  $d$  is

$$\tilde{\mathbf{h}} = \operatorname{argmax}_{\mathbf{h} \in \{0,1\}^n} \prod_{i=1}^n p_i^{\mathbf{h}_i}(d) = \left( \operatorname{argmax}_{h \in \{0,1\}} p_i^h(d) \right)_{i=1}^n.$$

Given  $\tilde{\mathbf{h}}$ , the second best hypothesis can be obtained by changing the label on the least probable classification, i.e., associated with the index  $j = \operatorname{argmin}_{i=1, \dots, n} p_i^{\tilde{\mathbf{h}}_i}(d)$ . By storing the probability of the  $k-1$  most probable configurations, the next  $k$  best hypotheses can be efficiently generated.

## 3 Structural Kernels for Reranking Hierarchical Classification

In this section we describe our hypothesis reranker. The main idea is to represent the hypotheses as a tree structure, naturally derived from the hierarchy and then to use tree kernels to encode such a structural description in a learning algorithm. For this purpose, we describe our hypothesis representation, kernel methods and the kernel-based approach to preference reranking.

### 3.1 Encoding hypotheses in a tree

Once hypotheses are generated, we need a representation from which the dependencies between the dif-

<sup>1</sup>We used the conversion of margin into probability provided by LIBSVM.

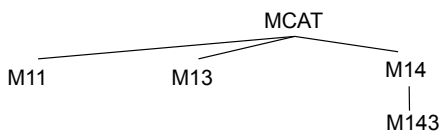


Figure 3: A compact representation of the hypothesis in Fig. 2.

ferent nodes of the hierarchy can be learned. Since we do not know in advance which are the important dependencies and not even the scope of the interaction between the different structure subparts, we rely on automatic feature engineering via structural kernels. For this paper, we consider tree-shaped hierarchies so that tree kernels, e.g. (Collins and Duffy, 2002; Moschitti, 2006a), can be applied.

In more detail, we focus on the Reuters categorization scheme. For example, Figure 1 shows a subhierarchy of the *Markets* (MCAT) category and its subcategories: *Equity Markets* (M11), *Bond Markets* (M12), *Money Markets* (M13) and *Commodity Markets* (M14). These also have subcategories: *Interbank Markets* (M131), *Forex Markets* (M132), *Soft Commodities* (M141), *Metals Trading* (M142) and *Energy Markets* (M143).

As the input of our reranker, we can simply use a tree representing the hierarchy above, marking the *negative* assignments of the current hypothesis in the node labels with “-”, e.g., -M142 means that the document was not classified in *Metals Trading*. For example, Figure 2 shows the representation of a classification hypothesis consisting in assigning the target document to the categories MCAT, M11, M13, M14 and M143.

Another more compact representation is the hierarchy tree from which all the nodes associated with a negative classification decision are removed. As only a small subset of nodes of the full hierarchy will be positively classified the tree will be much smaller. Figure 3 shows the compact representation of the hypothesis in Fig. 2. The next sections describe how to exploit these kinds of representations.

### 3.2 Structural Kernels

In kernel-based machines, both learning and classification algorithms only depend on the inner product between instances. In several cases, this can be efficiently and implicitly computed by kernel functions by exploiting the following dual formulation:

$\sum_{i=1..l} y_i \alpha_i \phi(o_i) \phi(o) + b = 0$ , where  $o_i$  and  $o$  are two objects,  $\phi$  is a mapping from the objects to feature vectors  $\vec{x}_i$  and  $\phi(o_i) \phi(o) = K(o_i, o)$  is a kernel function implicitly defining such a mapping. In case of structural kernels,  $K$  determines the shape of the substructures describing the objects above. The most general kind of kernels used in NLP are string kernels, e.g. (Shawe-Taylor and Cristianini, 2004), the Syntactic Tree Kernels (Collins and Duffy, 2002) and the Partial Tree Kernels (Moschitti, 2006a).

#### 3.2.1 String Kernels

The String Kernels (SK) that we consider count the number of subsequences shared by two strings of symbols,  $s_1$  and  $s_2$ . Some symbols during the matching process can be skipped. This modifies the weight associated with the target substrings as shown by the following SK equation:

$$SK(s_1, s_2) = \sum_{u \in \Sigma^*} \phi_u(s_1) \cdot \phi_u(s_2) = \sum_{u \in \Sigma^*} \sum_{\vec{I}_1: u=s_1[\vec{I}_1]} \sum_{\vec{I}_2: u=s_2[\vec{I}_2]} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)}$$

where,  $\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n$  is the set of all strings,  $\vec{I}_1$  and  $\vec{I}_2$  are two sequences of indexes  $\vec{I} = (i_1, \dots, i_{|u|})$ , with  $1 \leq i_1 < \dots < i_{|u|} \leq |s|$ , such that  $u = s_{i_1} \dots s_{i_{|u|}}$ ,  $d(\vec{I}) = i_{|u|} - i_1 + 1$  (distance between the first and last character) and  $\lambda \in [0, 1]$  is a decay factor.

It is worth noting that: (a) longer subsequences receive lower weights; (b) some characters can be omitted, i.e. gaps; (c) gaps determine a weight since the exponent of  $\lambda$  is the number of characters and gaps between the first and last character; and (c) the complexity of the SK computation is  $O(mnp)$  (Shawe-Taylor and Cristianini, 2004), where  $m$  and  $n$  are the lengths of the two strings, respectively and  $p$  is the length of the largest subsequence we want to consider.

In our case, given a hypothesis represented as a tree like in Figure 2, we can visit it and derive a linearization of the tree. SK applied to such a node sequence can derive useful dependencies between category nodes. For example, using the Breadth First Search on the compact representation, we get the sequence [MCAT, M11, M13, M14, M143], which generates the subsequences, [MCAT, M11], [MCAT, M11, M13, M14], [M11, M13, M143], [M11, M13, M143] and so on.

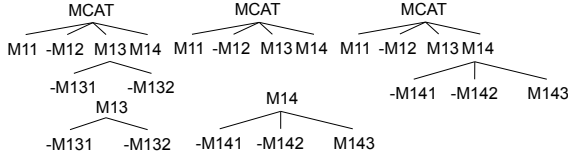


Figure 4: The tree fragments of the hypothesis in Fig. 2 generated by STK

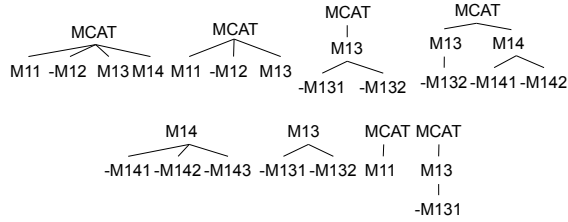


Figure 5: Some tree fragments of the hypothesis in Fig. 2 generated by PTK

### 3.2.2 Tree Kernels

Convolution Tree Kernels compute the number of common substructures between two trees  $T_1$  and  $T_2$  without explicitly considering the whole fragment space. For this purpose, let the set  $\mathcal{F} = \{f_1, f_2, \dots, f_{|\mathcal{F}|}\}$  be a tree fragment space and  $\chi_i(n)$  be an indicator function, equal to 1 if the target  $f_i$  is rooted at node  $n$  and equal to 0 otherwise. A tree-kernel function over  $T_1$  and  $T_2$  is  $TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$ ,  $N_{T_1}$  and  $N_{T_2}$  are the sets of the  $T_1$ 's and  $T_2$ 's nodes, respectively and  $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} \chi_i(n_1) \chi_i(n_2)$ . The latter is equal to the number of common fragments rooted in the  $n_1$  and  $n_2$  nodes. The  $\Delta$  function determines the richness of the kernel space and thus different tree kernels. Hereafter, we consider the equation to evaluate STK and PTK.<sup>2</sup>

**Syntactic Tree Kernels (STK)** To compute STK, it is enough to compute  $\Delta_{STK}(n_1, n_2)$  as follows (recalling that since it is a syntactic tree kernels, each node can be associated with a production rule): (i) if the productions at  $n_1$  and  $n_2$  are different then  $\Delta_{STK}(n_1, n_2) = 0$ ; (ii) if the productions at  $n_1$  and  $n_2$  are the same, and  $n_1$  and  $n_2$  have only leaf children then  $\Delta_{STK}(n_1, n_2) = \lambda$ ; and (iii) if the productions at  $n_1$  and  $n_2$  are the same, and  $n_1$  and  $n_2$  are not pre-terminals then  $\Delta_{STK}(n_1, n_2) = \lambda \prod_{j=1}^{l(n_1)} (1 + \Delta_{STK}(c_{n_1}^j, c_{n_2}^j))$ , where  $l(n_1)$  is the

<sup>2</sup>To have a similarity score between 0 and 1, a normalization in the kernel space, i.e.  $\frac{TK(T_1, T_2)}{\sqrt{TK(T_1, T_1) \times TK(T_2, T_2)}}$  is applied.

number of children of  $n_1$  and  $c_n^j$  is the  $j$ -th child of the node  $n$ . Note that, since the productions are the same,  $l(n_1) = l(n_2)$  and the computational complexity of STK is  $O(|N_{T_1}| |N_{T_2}|)$  but the average running time tends to be linear, i.e.  $O(|N_{T_1}| + |N_{T_2}|)$ , for natural language syntactic trees (Moschitti, 2006a; Moschitti, 2006b).

Figure 4 shows the five fragments of the hypothesis in Figure 2. Such fragments satisfy the constraint that each of their nodes includes all or none of its children. For example,  $[M13 [-M131 -M132]]$  is an STF, which has two non-terminal symbols,  $-M131$  and  $-M132$ , as leaves while  $[M13 [-M131]]$  is not an STF.

**The Partial Tree Kernel (PTK)** The computation of PTK is carried out by the following  $\Delta_{PTK}$  function: if the labels of  $n_1$  and  $n_2$  are different then  $\Delta_{PTK}(n_1, n_2) = 0$ ; else  $\Delta_{PTK}(n_1, n_2) =$

$$\mu \left( \lambda^2 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1) = l(\vec{I}_2)} \lambda^{d(\vec{I}_1) + d(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta_{PTK}(c_{n_1}(\vec{I}_{1j}), c_{n_2}(\vec{I}_{2j})) \right)$$

where  $d(\vec{I}_1) = \vec{I}_{1l(\vec{I}_1)} - \vec{I}_{11}$  and  $d(\vec{I}_2) = \vec{I}_{2l(\vec{I}_2)} - \vec{I}_{21}$ . This way, we penalize both larger trees and child subsequences with gaps. PTK is more general than STK as if we only consider the contribution of shared subsequences containing all children of nodes, we implement STK. The computational complexity of PTK is  $O(p\rho^2 |N_{T_1}| |N_{T_2}|)$  (Moschitti, 2006a), where  $p$  is the largest subsequence of children that we want consider and  $\rho$  is the maximal out-degree observed in the two trees. However the average running time again tends to be linear for natural language syntactic trees (Moschitti, 2006a).

Given a target  $T$ , PTK can generate any subset of connected nodes of  $T$ , whose edges are in  $T$ . For example, Fig. 5 shows the tree fragments from the hypothesis of Fig. 2. Note that each fragment captures dependencies between different categories.

### 3.3 Preference reranker

When training a reranker model, the task of the machine learning algorithm is to learn to select the best candidate from a given set of hypotheses. To use SVMs for training a reranker, we applied Preference Kernel Method (Shen et al., 2003). The reduction method from ranking tasks to binary classification is an active research area; see for instance (Balcan et al., 2008) and (Ailon and Mohri, 2010).

Category	Child-free				Child-full			
	Train	Train1	Train2	TEST	Train	Train1	Train2	TEST
C152	837	370	467	438	837	370	467	438
GPOL	723	357	366	380	723	357	366	380
M11	604	309	205	311	604	309	205	311
..	..	..	..	..	..	..	..	..
C31	313	163	150	179	531	274	257	284
E41	191	89	95	102	223	121	102	118
GCAT	345	177	168	173	3293	1687	1506	1600
..	..	..	..	..	..	..	..	..
E31	11	4	7	6	32	21	11	19
M14	96	49	47	58	1175	594	581	604
G15	5	4	1	0	290	137	153	146
Total: 103	10,000	5,000	5,000	5,000	10,000	5,000	5,000	5,000

Table 1: Instance distributions of RCV1: the most populated categories are on the top, the medium sized ones follow and the smallest ones are at the bottom. There are some difference between child-free and child-full setting since for the former, from each node, we removed all the documents in its children.

In the Preference Kernel approach, the reranking problem – learning to pick the correct candidate  $h_1$  from a candidate set  $\{h_1, \dots, h_k\}$  – is reduced to a binary classification problem by creating *pairs*: positive training instances  $\langle h_1, h_2 \rangle, \dots, \langle h_1, h_k \rangle$  and negative instances  $\langle h_2, h_1 \rangle, \dots, \langle h_k, h_1 \rangle$ . This training set can then be used to train a binary classifier. At classification time, pairs are not formed (since the correct candidate is not known); instead, the standard one-versus-all binarization method is still applied.

The kernels are then engineered to implicitly represent the *differences* between the objects in the pairs. If we have a valid kernel  $K$  over the candidate space  $\mathcal{T}$ , we can construct a preference kernel  $P_K$  over the space of pairs  $\mathcal{T} \times \mathcal{T}$  as follows:  $P_K(x, y) =$

$$P_K(\langle x_1, x_2 \rangle, \langle y_1, y_2 \rangle) = K(x_1, y_1) + K(x_2, y_2) - K(x_1, y_2) - K(x_2, y_1), \quad (1)$$

where  $x, y \in \mathcal{T} \times \mathcal{T}$ . It is easy to show (Shen et al., 2003) that  $P_K$  is also a valid Mercer’s kernel. This makes it possible to use kernel methods to train the reranker.

We explore innovative kernels  $K$  to be used in Eq. 1:

$K_J = p(x_1) \times p(y_1) + S$ , where  $p(\cdot)$  is the global joint probability of a target hypothesis and  $S$  is a structural kernel, i.e., SK, STK and PTK.

$K_P = \vec{x}_1 \cdot \vec{y}_1 + S$ , where  $\vec{x}_1 = \{p(x_1, j)\}_{j \in x_1}$ ,  $\vec{y}_1 = \{p(y_1, j)\}_{j \in y_1}$ ,  $p(t, n)$  is the classification probability of the node (category)  $n$  in the

$F_1$	BL	BOL	SK	STK	PTK
Micro- $F_1$	0.769	0.771	0.786	0.790	0.790
Macro- $F_1$	0.539	0.541	0.542	0.547	0.560

Table 2: Comparison of rerankers using different kernels, child-full setting ( $K_J$  model).

$F_1$	BL	BOL	SK	STK	PTK
Micro- $F_1$	0.640	0.649	0.653	0.677	0.682
Macro- $F_1$	0.408	0.417	0.431	0.447	0.447

Table 3: Comparison of rerankers using different kernels, child-free setting ( $K_J$  model).

tree  $t \in \mathcal{T}$  and  $S$  is again a structural kernel, i.e., SK, STK and PTK.

For comparative purposes, we also use for  $S$  a linear kernel over the bag-of-labels (BOL). This is supposed to capture non-structural dependencies between the category labels.

## 4 Experiments

The aim of the experiments is to demonstrate that our reranking approach can introduce semantic dependencies in the hierarchical classification model, which can improve accuracy. For this purpose, we show that several reranking models based on tree kernels improve the classification based on the *flat* one-vs.-all approach. Then, we analyze the efficiency of our models, demonstrating their applicability.

### 4.1 Setup

We used two full hierarchies, TOPICS and INDUSTRY of Reuters Corpus Volume 1 (RCV1)<sup>3</sup> TC cor-

<sup>3</sup>[trec.nist.gov/data/reuters/reuters.html](http://trec.nist.gov/data/reuters/reuters.html)

pus. For most experiments, we randomly selected two subsets of 10k and 5k of documents for training and testing from the total 804,414 Reuters news from TOPICS by still using all the 103 categories organized in 5 levels (hereafter SAM). The distribution of the data instances of some of the different categories in such samples can be observed in Table 1. The training set is used for learning the binary classifiers needed to build the multiclass-classifier (MCC). To compare with previous work we also considered the Lewis’ split (Lewis et al., 2004), which includes 23,149 news for training and 781,265 for testing.

Additionally, we carried out some experiments on INDUSTRY data from RCV1. This contains 352,361 news assigned to 365 categories, which are organized in 6 levels. The Lewis’ split for INDUSTRY includes 9,644 news for training and 342,117 for testing. We used the above datasets with two different settings: the *child-free* setting, where we removed all the document belonging to the child nodes from the parent nodes, and the normal setting which we refer to as *child-full*.

To implement the baseline model, we applied the state-of-the-art method used by (Lewis et al., 2004) for RCV1, i.e.: SVMs with the default parameters (trade-off and cost factor = 1), linear kernel, normalized vectors, stemmed bag-of-words representation,  $\log(TF + 1) \times IDF$  weighting scheme and stop list<sup>4</sup>. We used the LIBSVM<sup>5</sup> implementation, which provides a probabilistic outcome for the classification function. The classifiers are combined using the one-vs.-all approach, which is also state-of-the-art as argued in (Rifkin and Klautau, 2004). Since the task requires us to assign multiple labels, we simply collect the decisions of the  $n$  classifiers: this constitutes our MCC baseline.

Regarding the reranker, we divided the training set in two chunks of data: Train1 and Train2. The binary classifiers are trained on Train1 and tested on Train2 (and vice versa) to generate the hypotheses on Train2 (Train1). The union of the two sets constitutes the training data for the reranker. We imple-

<sup>4</sup>We have just a small difference in the number of tokens, i.e., 51,002 vs. 47,219 but this is both not critical and rarely achievable because of the diverse stop lists or tokenizers.

<sup>5</sup><http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

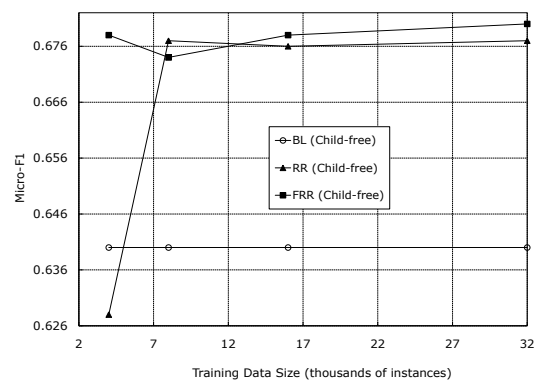


Figure 6: Learning curves of the reranking models using STK in terms of MicroAverage-F1, according to increasing training set (child-free setting).

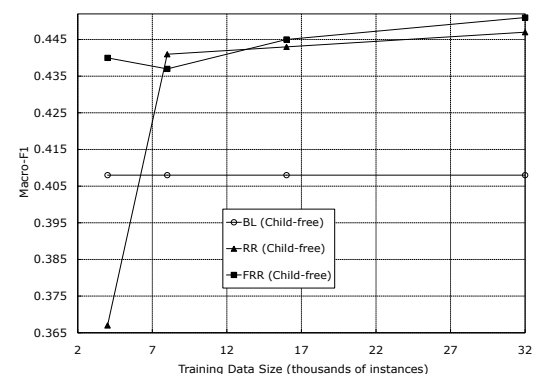


Figure 7: Learning curves of the reranking models using STK in terms of MacroAverage-F1, according to increasing training set (child-free setting).

mented two rerankers: RR, which use the representation of hypotheses described in Fig. 2; and FRR, i.e., fast RR, which uses the compact representation described in Fig. 3.

The rerankers are based on SVMs and the Preference Kernel ( $P_K$ ) described in Sec. 1 built on top of SK, STK or PTK (see Section 3.2.2). These are applied to the tree-structured hypotheses. We trained the rerankers using SVM-light-TK<sup>6</sup>, which enables the use of structural kernels in SVM-light (Joachims, 1999). This allows for applying kernels to pairs of trees and combining them with vector-based kernels. Again we use default parameters to facilitate replicability and preserve generality. The rerankers always use 8 best hypotheses.

All the performance values are provided by means of Micro- and Macro-Average F1, evaluated on test

<sup>6</sup>[disi.unitn.it/moschitti/Tree-Kernel.htm](http://disi.unitn.it/moschitti/Tree-Kernel.htm)



Cat.	Child-free			Child-full		
	BL	$K_J$	$K_P$	BL	$K_J$	$K_P$
C152	0.671	0.700	0.771	0.671	0.729	0.745
GPOL	0.660	0.695	0.743	0.660	0.680	0.734
M11	0.851	0.891	0.901	0.851	0.886	0.898
..	..	..	..	..	..	..
C31	0.225	0.311	0.446	0.356	0.421	0.526
E41	0.643	0.714	0.719	0.776	0.791	0.806
GCAT	0.896	0.908	0.917	0.908	0.916	0.926
..	..	..	..	..	..	..
E31	0.444	0.600	0.600	0.667	0.765	0.688
M14	0.591	0.600	0.575	0.887	0.897	0.904
G15	0.250	0.222	0.250	0.823	0.806	0.826
103 cat.						
Mi-F1	0.640	0.677	0.731	0.769	0.794	0.815
Ma-F1	0.408	0.447	0.507	0.539	0.567	0.590

Table 4: F1 of some binary classifiers along with the Micro and Macro-Average F1 over all 103 categories of RCV1, 8 hypotheses and 32k of training data for rerankers using STK.

data over all categories (103 or 363). Additionally, the F1 of some binary classifiers are reported.

## 4.2 Classification Accuracy

In the first experiments, we compared the different kernels using the  $K_J$  combination (which exploits the joint hypothesis probability, see Sec. 3.3) on SAM. Tab. 2 shows that the baseline (state-of-the-art flat model) is largely improved by all rerankers. BOL cannot capture the same dependencies as the structural kernels. In contrast, when we remove the dependencies generated by shared documents between a node and its descendants (child-free setting) BOL improves on BL. Very interestingly, TK and PTK in this setting significantly improves on SK suggesting that the hierarchical structure is more important than the sequential one.

To study how much data is needed for the reranker, the figures 6 and 7 report the Micro and Macro Average F1 of our rerankers over 103 categories, according to different sets of training data. This time,  $K_J$  is applied to only STK. We note that (i) a few thousands of training examples are enough to deliver most of the RR improvement; and (ii) the FRR produces similar results as standard RR. This is very interesting since, as it will be shown in the next section, the compact representation produces much faster models.

Table 4 reports the F1 of some individual categories as well as global performance. In these experiments we used STK in  $K_J$  and  $K_P$ . We note that

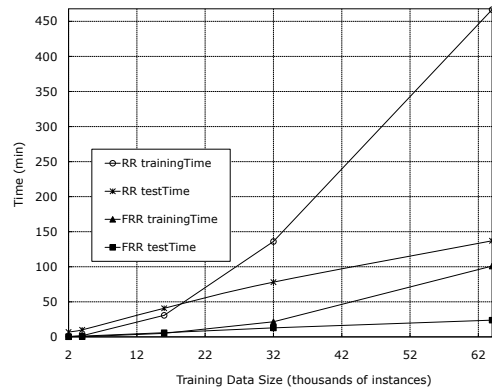


Figure 8: Training and test time of the rerankers trained on data of increasing size.

$K_P$  highly improves on the baseline on child-free setting by about 7.1 and 9.9 absolute percent points in Micro-and Macro-F1, respectively. Also the improvement on child-full is meaningful, i.e., 4.6 percent points. This is rather interesting as BOL (not reported in the table) achieved a Micro-average of 80.4% and a Macro-average of 57.2% when used in  $K_P$ , i.e., up to 2 points below STK. This means that the use of probability vectors and combination with structural kernels is a very promising direction for reranker design.

To definitely assess the benefit of our rerankers we tested them on the Lewis’ split of two different datasets of RCV1, i.e., TOPIC and INDUSTRY. Table 5 shows impressive results, e.g., for INDUSTRY, the improvement is up to 5.2 percent points. We carried out statistical significance tests, which certified the significance at 99%. This was expected as the size of the Lewis’ test sets is in the order of several hundreds thousands.

Finally, to better understand the potential of reranking, Table 6 shows the oracle performance with respect to the increasing number of hypotheses. The outcome clearly demonstrates that there is large margin of improvement for the rerankers.

## 4.3 Running Time

To study the applicability of our rerankers, we have analyzed both the training and classification time. Figure 8 shows the minutes required to train the different models as well as to classify the test set according to data of increasing size.

It can be noted that the models using the compact hypothesis representation are much faster than those

F1	Topic					Industry				
	BL (Lewis)	BL (Ours)	$K_J$ (BOL)	$K_J$	$K_P$	BL (Lewis)	BL (Ours)	$K_J$ (BOL)	$K_J$	$K_P$
Micro-F1	0.816	0.815	0.818	0.827	0.849	0.512	0.562	0.566	0.576	0.628
Macro-F1	0.567	0.566	0.571	0.590	0.615	0.263	0.289	0.243	0.314	0.341

Table 5: Comparison between rankers using STK or BOL (when indicated) with the  $K_J$  and  $K_P$  schema. 32k examples are used for training the rerankers with child-full setting.

$k$	Micro- $F_1$	Macro- $F_1$
1	0.640	0.408
2	0.758	0.504
4	0.821	0.566
8	0.858	0.610
16	0.898	0.658

Table 6: Oracle performance according to the number of hypotheses (child-free setting).

using the complete hierarchy as representation, i.e., up to five times in training and eight time in testing. This is not surprising as, in the latter case, each kernel evaluation requires to perform tree kernel evaluation on trees of 103 nodes. When using the compact representation the number of nodes is upper-bounded by the maximum number of labels per documents, i.e., 6, times the depth of the hierarchy, i.e., 5 (the positive classification on the leaves is the worst case). Thus, the largest tree would contain 30 nodes. However, we only have 1.82 labels per document on average, therefore the trees have an average size of only about 9 nodes.

## 5 Related Work

Tree and sequence kernels have been successfully used in many NLP applications, e.g.: parse reranking and adaptation (Collins and Duffy, 2002; Shen et al., 2003; Toutanova et al., 2004; Kudo et al., 2005; Titov and Henderson, 2006), chunking and dependency parsing (Kudo and Matsumoto, 2003; Daumé III and Marcu, 2004), named entity recognition (Cumby and Roth, 2003), text categorization (Cancedda et al., 2003; Gliozzo et al., 2005) and relation extraction (Zelenko et al., 2002; Bunescu and Mooney, 2005; Zhang et al., 2006).

To our knowledge, ours is the first work exploring structural kernels for reranking hierarchical text categorization hypotheses. Additionally, there is a substantial lack of work exploring reranking for hierarchical text categorization. The work mostly related to ours is (Rousu et al., 2006) as they directly encoded global dependencies in a gradient descent learning approach. This kind of algorithm is less efficient than ours so they could experiment

with only the CCAT subhierarchy of RCV1, which only contains 34 nodes. Other relevant work such as (McCallum et al., 1998) and (Dumais and Chen, 2000) uses a rather different datasets and a different idea of dependencies based on feature distributions over the linked categories. An interesting method is SVM-struct (Tsochantaridis et al., 2005), which has been applied to model dependencies expressed as category label subsets of flat categorization schemes but no solution has been attempted for hierarchical settings. The approaches in (Finley and Joachims, 2007; Riezler and Vasserman, 2010; Lavergne et al., 2010) can surely be applied to model dependencies in a tree, however, they need that feature templates are specified in advance, thus the meaningful dependencies must be already known. In contrast, kernel methods allow for automatically generating all possible dependencies and reranking can efficiently encode them.

## 6 Conclusions

In this paper, we have described several models for reranking the output of an MCC based on SVMs and structural kernels, i.e., SK, STK and PTK. We have proposed a simple and efficient algorithm for hypothesis generation and their kernel-based representations. The latter are exploited by SVMs using preference kernels to automatically derive features from the hypotheses. When using tree kernels such features are tree fragments, which can encode complex semantic dependencies between categories. We tested our rerankers on the entire well-known RCV1. The results show impressive improvement on the state-of-the-art flat TC models, i.e., 3.3 absolute percent points on the Lewis' split (same setting) and up to 10 absolute points on samples using child-free setting.

**Acknowledgements** This research is partially supported by the EC FP7/2007-2013 under the grants: 247758 (ETERNALS), 288024 (LiMOSINE) and 231126 (LIVINGKNOWLEDGE). Many thanks to the reviewers for their valuable suggestions.

## References

- Nir Ailon and Mehryar Mohri. 2010. Preference-based learning to rank. *Machine Learning*.
- Maria-Florina Balcan, Nikhil Bansal, Alina Beygelzimer, Don Coppersmith, John Langford, and Gregory B. Sorkin. 2008. Robust reductions from ranking to classification. *Machine Learning*, 72(1-2):139–153.
- Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of HLT and EMNLP*, pages 724–731, Vancouver, British Columbia, Canada, October.
- Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean Michel Renders. 2003. Word sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082.
- Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL'02*, pages 263–270.
- Chad Cumby and Dan Roth. 2003. On kernel methods for relational learning. In *Proceedings of ICML 2003*.
- Hal Daumé III and Daniel Marcu. 2004. Np bracketing by maximum entropy tagging and SVM reranking. In *Proceedings of EMNLP'04*.
- Susan T. Dumais and Hao Chen. 2000. Hierarchical classification of web content. In Nicholas J. Belkin, Peter Ingwersen, and Mun-Kew Leong, editors, *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Athens, GR. ACM Press, New York, US.
- T. Finley and T. Joachims. 2007. Parameter learning for loopy markov random fields with structural support vector machines. In *ICML Workshop on Constrained Optimization and Structured Output Spaces*.
- Alfio Gliozzo, Claudio Giuliano, and Carlo Strapparava. 2005. Domain kernels for word sense disambiguation. In *Proceedings of ACL'05*, pages 403–410.
- Thorsten Joachims. 1999. Making large-scale SVM learning practical. *Advances in Kernel Methods – Support Vector Learning*, 13.
- Taku Kudo and Yuji Matsumoto. 2003. Fast methods for kernel-based text analysis. In *Proceedings of ACL'03*.
- Taku Kudo, Jun Suzuki, and Hideki Isozaki. 2005. Boosting-based parse reranking with subtree features. In *Proceedings of ACL'05*.
- T. Lavergne, O. Cappé, and F. Yvon. 2010. Practical very large scale CRFs. In *Proc. of ACL*, pages 504–513.
- D. D. Lewis, Y. Yang, T. Rose, and F. Li. 2004. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, (5):361–397.
- Andrew McCallum, Ronald Rosenfeld, Tom M. Mitchell, and Andrew Y. Ng. 1998. Improving text classification by shrinkage in a hierarchy of classes. In *ICML*, pages 359–367.
- Alessandro Moschitti. 2006a. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of ECML'06*.
- Alessandro Moschitti. 2006b. Making tree kernels practical for natural language learning. In *Proceedings of EACL'06*.
- S. Riezler and A. Vasserman. 2010. Incremental feature selection and l1 regularization for relaxed maximum-entropy modeling. In *EMNLP*.
- Ryan Rifkin and Aldebaro Klautau. 2004. In defense of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141, December.
- Juho Rousu, Craig Saunders, Sandor Szedmak, and John Shawe-Taylor. 2006. Kernel-based learning of hierarchical multilabel classification models. *The Journal of Machine Learning Research*, (7):1601–1626.
- John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.
- Libin Shen, Anoop Sarkar, and Aravind k. Joshi. 2003. Using LTAG Based Features in Parse Reranking. In *Empirical Methods for Natural Language Processing (EMNLP)*, pages 89–96, Sapporo, Japan.
- Ivan Titov and James Henderson. 2006. Porting statistical parsers with data-defined kernels. In *Proceedings of CoNLL-X*.
- Kristina Toutanova, Penka Markova, and Christopher Manning. 2004. The Leaf Path Projection View of Parse Trees: Exploring String Kernels for HPSG Parse Selection. In *Proceedings of EMNLP 2004*.
- Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *J. Machine Learning Reserach.*, 6:1453–1484, December.
- Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2002. Kernel methods for relation extraction. In *Proceedings of EMNLP-ACL*, pages 181–201.
- Min Zhang, Jie Zhang, and Jian Su. 2006. Exploring Syntactic Features for Relation Extraction using a Convolution tree kernel. In *Proceedings of NAACL*.

# Attacking Parsing Bottlenecks with Unlabeled Data and Relevant Factorizations

Emily Pitler

Computer and Information Science  
University of Pennsylvania  
Philadelphia, PA 19104  
epitler@seas.upenn.edu

## Abstract

*Prepositions* and *conjunctions* are two of the largest remaining bottlenecks in parsing. Across various existing parsers, these two categories have the lowest accuracies, and mistakes made have consequences for downstream applications. Prepositions and conjunctions are often assumed to depend on lexical dependencies for correct resolution. As lexical statistics based on the training set only are sparse, unlabeled data can help ameliorate this sparsity problem. By including unlabeled data features into a factorization of the problem which matches the representation of prepositions and conjunctions, we achieve a new state-of-the-art for English dependencies with 93.55% correct attachments on the current standard. Furthermore, conjunctions are attached with an accuracy of 90.8%, and prepositions with an accuracy of 87.4%.

## 1 Introduction

*Prepositions* and *conjunctions* are two large remaining bottlenecks in parsing. Across various existing parsers, these two categories have the lowest accuracies, and mistakes made on these have consequences for downstream applications. Machine translation is sensitive to parsing errors involving prepositions and conjunctions, because in some languages different attachment decisions in the parse of the source language sentence produce different translations. Preposition attachment mistakes are particularly bad when translating into Japanese (Schwartz et al., 2003) which uses a different post-position for different attachments; conjunction mis-

takes can cause word ordering mistakes when translating into Chinese (Huang, 1983).

Prepositions and conjunctions are often assumed to depend on *lexical dependencies* for correct resolution (Jurafsky and Martin, 2008). However, lexical statistics based on the training set only are typically sparse and have only a small effect on overall parsing performance (Gildea, 2001). *Unlabeled data* can help ameliorate this sparsity problem. Backing off to cluster membership features (Koo et al., 2008) or by using association statistics from a larger corpus, such as the web (Bansal and Klein, 2011; Zhou et al., 2011), have both improved parsing.

Unlabeled data has been shown to improve the accuracy of conjunctions within complex noun phrases (Pitler et al., 2010; Bergsma et al., 2011). However, it has so far been less effective within full parsing — while first-order web-scale counts noticeably improved overall parsing in Bansal and Klein (2011), the accuracy on conjunctions actually decreased when the web-scale features were added (Table 4 in that paper).

In this paper we show that unlabeled data *can* help prepositions and conjunctions, *provided that the dependency representation is compatible with how the parsing problem is decomposed for learning and inference*. By incorporating unlabeled data into factorizations which capture the relevant dependencies for prepositions and conjunctions, we produce a parser for English which has an unlabeled attachment accuracy of 93.5%, over an 18% reduction in error over the best previously published parser (Bansal and Klein, 2011) on the current standard for dependency parsing. The best model for conjunctions at-

taches them with 90.8% accuracy (42.5% reduction in error over MSTParser), and the best model for prepositions with 87.4% accuracy (18.2% reduction in error over MSTParser).

We describe the dependency representations of prepositions and conjunctions in Section 2. We discuss the implications of these representations for how learning and inference for parsing are decomposed (Section 3) and how unlabeled data may be used (Section 4). We then present experiments exploring the connection between representation, factorization, and unlabeled data in Sections 5 and 6.

## 2 Dependency Representations

A dependency tree is a rooted, directed tree (or arborecence), in which the vertices are the words in the sentence plus an artificial root node, and each edge  $(h, m)$  represents a directed dependency relation from the head  $h$  to the modifier  $m$ . Throughout this section, we will use  $Y$  to denote a particular parse tree, and  $(h, m) \in Y$  to denote a particular edge in  $Y$ .

The Wall Street Journal Penn Treebank (PTB) (Marcus et al., 1993) contains parsed constituency trees (where each sentence is represented as a context-free-grammar derivation). Dependency parsing requires a conversion from these constituency trees to dependency trees. The Treebank constituency trees left noun phrases (NPs) flat, although there have been subsequent projects which annotate the internal structure of noun phrases (Vadas and Curran, 2007; Weischedel et al., 2011). The presence or absence of these noun phrase internal annotations interacts with constituency-to-dependency conversion program in ways which have effects on conjunctions and prepositions.

We consider two such mapping regimes here:

1. PTB trees  $\rightarrow$  *Penn2Malt*<sup>1</sup>  $\rightarrow$  Dependencies
2. PTB trees patched with NP-internal annotations (Vadas and Curran, 2007)  $\rightarrow$  *pennconverter*<sup>2</sup>  $\rightarrow$  Dependencies

<sup>1</sup><http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html>

<sup>2</sup>Johansson and Nugues (2007) [http://nlp.cs.lth.se/software/treebank\\_converter/](http://nlp.cs.lth.se/software/treebank_converter/)

Regime (1) is very commonly done in papers which report dependency parsing experiments (e.g., (McDonald and Pereira, 2006; Nivre et al., 2007; Zhang and Clark, 2008; Huang and Sagae, 2010; Koo and Collins, 2010)). *Penn2Malt* uses the head finding table from Yamada and Matsumoto (2003).

Regime (2) is based on the recommendations of the two converter tools; as of the date of this writing, the *Penn2Malt* website says: “Penn2Malt has been superseded by the more sophisticated *pennconverter*, which we strongly recommend”. The *pennconverter* website “strongly recommends” patching the Treebank with the NP annotations of Vadas and Curran (2007). A version of *pennconverter* was used to prepare the data for the CoNLL Shared Tasks of 2007-2009, so the trees produced by Regime 2 are similar (but not identical)<sup>3</sup> to these shared tasks. As far as we are aware, Bansal and Klein (2011) is the only published work which uses both steps in Regime (2).

The dependency representations produced by Regime 2 are designed to be more useful for extracting semantics (Johansson and Nugues, 2007). The parsing attachment accuracy of MALTPARSER (Nivre et al., 2007) was lower using *pennconverter* than *Penn2Malt*, but using the output of MALTPARSER under the new format parses produces a much better semantic role labeler than using its output with *Penn2Malt* (Johansson and Nugues, 2007).

Figures 1 and 2 show how conjunctions and prepositions, respectively, are represented after the two different conversion processes. *These differences are not rare*—70.7% of conjunctions and 5.2% of prepositions in the development set have a different parent under the two conversion types. These representational differences have serious implications for how well various factorizations will be able to capture these two phenomena.

## 3 Implications of Representations on the Scope of Factorization

Parsing requires a) learning to score potential parse trees, and b) given a particular scoring function, finding the highest scoring tree according to that function. The number of potential trees for a sen-

<sup>3</sup>The CoNLL data does not include the NP annotations; it does include annotations of named entities (Weischedel and Brunstein, 2005) so had some internal NP edges.

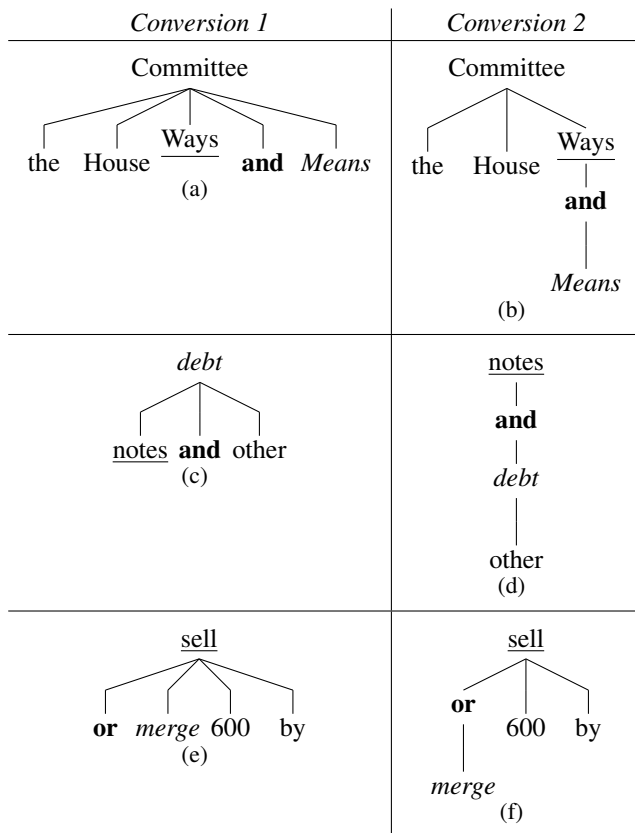


Figure 1: Examples of conjunctions: *the House Ways and Means Committee*, *notes and other debt*, and *sell or merge 600 by*. The conjunction is bolded, the left conjunct (in the linear order of the sentence) is underlined, and the right conjunct is italicized.

tence is exponential, so parsing is made tractable by decomposing the problem into a set of local substructures which can be combined using dynamic programming. Four possible factorizations are: single edges (edge-based), pairs of edges which share a parent (siblings), pairs of edges where the child of one is the parent of the other (grandparents), and triples of edges where the child of one is the parent of two others (grandparent+sibling). In this section, we discuss these factorizations and their relevance to conjunction and preposition representations.

### 3.1 Edge-based Scoring

One possible factorization corresponds to first-order parsing, in which the score of a parse tree  $Y$  decomposes completely across the edges in the tree:

$$S(Y) = \sum_{(h,m) \in Y} S(h,m) \quad (1)$$

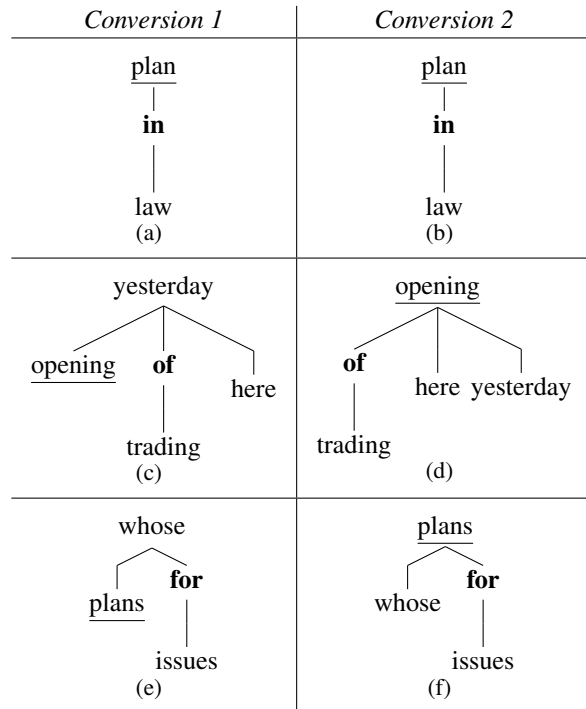


Figure 2: Examples of prepositions: *plan in the S&L bailout law*, *opening of trading here yesterday*, and *whose plans for major rights issues*. The preposition is bolded and the (semantic) head is underlined.

**Conjunctions:** Under Conversion 1, we can see three different representations of conjunctions in Figures 1(a), 1(c), and 1(e). Under edge-based scoring, the conjunction would be scored along with *neither* of its conjuncts in 1(a). In Figure 1(c), the conjunction is scored along with its right conjunct only; in figure 1(e) along with its left conjunct only. The inconsistency here is likely to make learning more difficult, as what is learned is split across these three cases. Furthermore, the conjunction is connected with an edge to either zero or one of its two arguments; at least one of the arguments is completely ignored in terms of scoring the conjunction.

In Figures 1(c) and 1(e), the words being conjoined are connected to *each other* by an edge. This overloads the meaning of an edge; an edge indicates both a head-modifier relationship and a conjunction relationship. For example, compare the two natural phrases *dogs and cats* and *really nice dogs*. *dogs* and *cats* are a good pair to conjoin, but *cats* is not a good modifier for *dogs*, so there is a tension when scoring an edge like *(dogs, cats)*: it should get a high score

when actually indicating a conjunction and low otherwise. (*nice, really*) shows the opposite pattern—*really* is a good modifier for *nice*, but *nice* and *really* are not two words which should be conjoined. This may be partially compensated for by including features about the surrounding words (McDonald et al., 2005), but any feature templates which would be identical across the two contexts will be in tension.

In Figures 1(b), 1(d) and 1(f), the conjunction participates in a directed edge with each of the conjuncts. Thus, in edge-based scoring, at least under Conversion 2 neither of the conjuncts is being ignored; however, the factorization scores each edge independently, so how compatible these two conjuncts are with each other cannot be included in the scoring of a tree.

**Prepositions:** For all of the examples in Figure 2, there is a directed edge from the head of the phrase that the preposition modifies to the preposition. Differences in head finding rules account for the differences in preposition representations. In the second example, the first conversion scheme chooses *yesterday* as the head of the overall NP, resulting in the edge *yesterday*→*of*, while the second conversion scheme ignores temporal phrases when finding the head, resulting in the more semantically meaningful *opening*→*of*. Similarly, in the third example, the preposition *for* attaches to the pronoun *whose* in the first conversion scheme, while it attaches to the noun *plans* in the second.

With edge-based scoring, the object is not accessible when scoring where the preposition should attach, and PP-attachment is known to depend on the object of the preposition (Hindle and Rooth, 1993).

### 3.2 Sibling Scoring

Another alternative factorization is to score *siblings* as well as parent-child edges (McDonald and Pereira, 2006). Scores decompose as:

$$S(Y) = \sum_{(h, m, s)} S(h, m, s) \quad (2)$$

$$\left\{ (h, m, s) \mid \begin{array}{l} (h, m) \in Y, (h, s) \in Y, \\ (m, s) \in Sib(Y) \end{array} \right\}$$

where  $Sib(Y)$  is the set containing *ordered* and *adjacent* sibling pairs in  $Y$ : if  $(m, s) \in Sib(Y)$ , there must exist a shared parent  $h$  such that  $(h, m) \in Y$  and  $(h, s) \in Y$ ,  $m$  and  $s$  must be on the same side of  $h$ ,  $m$  must be closer to  $h$  than  $s$  in the linear order

of the sentence, and there must not exist any other children of  $h$  in between  $m$  and  $s$ .

Under this factorization, two of the three examples in Conversion 1 (and none of the examples in Conversion 2) in Figure 1 now include the conjunction and both conjuncts in the same score (Figures 1(c) and 1(e)). The scoring for head-modifier dependencies and conjunction dependencies are again being overloaded: (*debt, notes, and*) and (*debt, and, other*) are both sibling parts in Figure 1(c), yet only one of them represents a conjunction. The position of the conjunction in the sibling is not enough to determine whether one is scoring a true conjunction relation or just the conjunction and a different sibling; in 1(c) the conjunction is on the right of its sibling argument, while in 1(e) the conjunction is on the left.

For none of the other preposition or conjunction examples does a sibling factorization bring more of the arguments into the scope of what is scored along with the preposition/conjunction. Sibling scoring may have some benefit in that prepositions/conjunctions should have only one argument, so for prepositions (under both conversions) and conjunctions (under Conversion 2), the model can learn to disprefer the existence of any siblings and thus enforce choosing a single child.

### 3.3 Grandparent Scoring

Another alternative over pairs of edges scores grandparents instead of siblings, with factorization:

$$S(Y) = \sum_{\{ (h, m, c) \mid (h, m) \in Y, (m, c) \in Y \}} S(h, m, c) \quad (3)$$

Under Conversion 2, we would expect this factorization to perform much better on conjunctions and prepositions than edge-based or sibling-based factorizations. Both conjunctions and prepositions are consistently represented by exactly one grandparent relation (with one relevant argument as the grandparent, the preposition/conjunction as the parent, and the other argument as the child), so this is the first factorization that has allowed the compatibility of the two arguments to affect the attachment of the preposition/conjunction.

Under Conversion 1, this factorization is particularly appropriate for prepositions, but would be unlikely to help conjunctions, which have no children.

### 3.4 Grandparent-Sibling Scoring

A further widening of the factorization takes grandparents and siblings simultaneously:

$$S(Y) = \sum_{(g, h, m, s)} S(g, h, m, s) \quad (4)$$
$$\left\{ (g, h, m, s) \mid \begin{array}{l} (g, h) \in Y, (h, m) \in Y, \\ (h, s) \in Y, (m, s) \in \text{Sib}(Y) \end{array} \right\}$$

For projective parsing, dynamic programming for this factorization was derived in Koo and Collins (2010) (Model 1 in that paper), and for non-projective parsing, dual decomposition was used for this factorization in Koo et al. (2010).

This factorization should combine all the benefits of the sibling and grandparent factorizations described above—for Conversion 1, sibling scoring may help conjunctions and grandparent scoring may help prepositions, and for Conversion 2, grandparent scoring should help both, while sibling scoring may or may not add some additional gains.

## 4 Using Unlabeled Data Effectively

Associations from unlabeled data have the potential to improve both conjunctions and prepositions. We predict that web counts which include both conjuncts (for conjunctions), or which include both the attachment site and the object of a preposition (for prepositions) will lead to the largest improvements.

For the phrase *dogs and cats*, edge-based counts would measure the associations between *dogs* and *and*, and *and* and *cats*, but never any web counts that include both *dogs* and *cats*. For the phrase *ate spaghetti with a fork*, edge-based scoring would not use any web counts involving both *ate* and *fork*.

We use *associations* rather than raw counts. The phrases *trading and transacting* versus *trading and what* provide an example of the difference between associations and counts. The phrase *trading and what* has a higher count than the phrase *trading and transacting*, but *trading and transacting* are more highly associated. In this paper, we use point-wise mutual information (PMI) to measure the strength of associations of words participating in potential conjunctions or prepositions.<sup>4</sup> For three words  $h, m, c$ , this is calculated with:

$$PMI(h, m, c) = \log \frac{P(h * m * c)}{P(h)P(m)P(c)} \quad (5)$$

<sup>4</sup>PMI can be unreliable when frequency counts are small (Church and Hanks, 1990), however the data used was thresholded, so all counts used are at least 10.

The probabilities are estimated using web-scale n-gram counts, which are looked up using the tools and web-scale n-grams described in Lin et al. (2010). Defining the joint probability using wild-cards (rather than the exact sequence  $h m c$ ) is crucially important, as determiners, adjectives, and other words may naturally intervene between the words of interest.

Approaches which cluster words (i.e., Koo et al. (2008)) are also designed to identify words which are semantically related. As manually labeled parsed data is sparse, this may help generalize across similar words. However, if edges are not connected to the semantic head, cluster-based methods may be less effective. For example, the choice of *yesterday* as the head of *opening of trading here yesterday* in Figure 2(c) or *whose* in 2(e) may make cluster-based features less useful than if the semantic heads were chosen (*opening* and *plans*, respectively).

## 5 Experiments

The previous section motivated the use of unlabeled data for attaching prepositions and conjunctions. We have also hypothesized that these features will be most effective when the *data representation* and the *learning representation* both capture relevant properties of prepositions and conjunctions. We predict that Conversion 2 and a factorization which includes grand-parent scoring will achieve the highest performance. In this section, we investigate the impact of unlabeled data on parsing accuracy using the two conversions and using each of the factorizations described in Section 3.1-3.4.

### 5.1 Unlabeled Data Feature Set

**Clusters:** We replicate the cluster-based features from Koo et al. (2008), which includes features over *all* edges  $(h, m)$ , grand-parent triples  $(h, m, c)$ , and parent sibling triples  $(h, m, s)$ . The features were all derived from the publicly available clusters produced by running the Brown clustering algorithm (Brown et al., 1992) over the BLLIP corpus with the Penn Treebank sentences excluded.<sup>5</sup>

Preposition and conjunction-inspired features (motivated by Section 4) are described below:

<sup>5</sup>[people.csail.mit.edu/maestro/papers/bllip-clusters.gz](http://people.csail.mit.edu/maestro/papers/bllip-clusters.gz)



**Web Counts:** For each set of words of interest, we compute the PMI between the words, and then include binary features for whether the mutual information is undefined, if it is negative, and whether it is greater than each positive integer.

For conjunctions, we only do this for triples of both conjunct and the conjunction (and if the conjunction is *and* or *or* and the two potential conjuncts are the same coarse grained part-of-speech). For prepositions, we consider only cases in which the parent is a noun or a verb and the child is a noun (this corresponds to the cases considered by Hindle and Rooth (1993) and others). Prepositions use association features to score both the triple (parent, preposition, child) and all pairs within that triple. The counts features are not used if all the words involved are stopwords. For the scope of this paper we use *only* the above counts related to prepositions and conjunctions.

## 5.2 Parser

We use the Model 1 version of *dpo3*, a state-of-the-art third-order dependency parser (Koo and Collins, 2010)<sup>6</sup>. We augment the feature set used with the web-counts-based features relevant to prepositions and conjunctions and the cluster-based features. The only other change to the parser’s existing feature set was the addition of binary features for the part-of-speech tag of the child of the root node, alone and conjoined with the tags of its children. For further details about the parser, see Koo and Collins (2010).

## 5.3 Experimental Set-up

Training was done on Section 2-21 of the Penn Treebank. Section 22 was used for development, and Section 23 for test. We use automatic part-of-speech tags for both training and testing (Ratnaparkhi, 1996). The set of potential edges was pruned using the marginals produced by a first-order parser trained using exponentiated gradient descent (Collins et al., 2008) as in Koo and Collins (2010). We train the full parser for 15 iterations of averaged perceptron training (Collins, 2002), choose the iteration with the best unlabeled attachment score (UAS) on the development set, and apply the model after that iteration to the test set.

<sup>6</sup><http://groups.csail.mit.edu/nlp/dpo3/>

We also ran MSTParser (McDonald and Pereira, 2006), the Berkeley constituency parser (Petrov and Klein, 2007), and the unmodified *dpo3* Model 1 (Koo and Collins, 2010) using Conversion 2 (the current recommendations) for comparison. Since the converted Penn Treebank now contains a few non-projective sentences, we ran both the projective and non-projective versions of the second order (sibling) MSTParser. The Berkeley parser was trained on the constituency trees of the PTB patched with Vadas and Curran (2007), and then the predicted parses were converted using *pennconverter*.

## 6 Results and Discussion

Table 1 shows the unlabeled attachment scores, complete sentence exact match accuracies, and the accuracies of conjunctions and prepositions under Conversion 2.<sup>7</sup> The incorporation of the unlabeled data features (clusters and web counts) into the *dpo3* parser yields a significantly better parser than *dpo3* alone (93.54 UAS versus 93.21)<sup>8</sup>, and is more than a 1.5% improvement over MSTParser.

### 6.1 Impact of Factorization

In all four metrics (attachment of all non-punctuation tokens, sentence accuracy, prepositions, and conjunctions), there is no significant difference between the version of the parser which uses the grandparent and sibling factorization (Grand+Sib) and the version which uses just the grandparent factorization (Grand). A parser which uses only grandparents (referred to as Model 0 in Koo and Collins (2010)) may therefore be preferable, as it contains far fewer parameters than a third-order parser.

While the grandparent factorization and the sibling factorization (Sib) are both “second-order” parsers, scoring up to two edges (involving three words) simultaneously, their results are quite different, with the sibling factorization scoring much worse. This is particularly notable in the conjunction case, where the sibling model is over 5% absolute worse in accuracy than the grandparent model.

<sup>7</sup>As is standard for English dependency parsing, five punctuation symbols :, , “ ”, and . are excluded from the results (Yamada and Matsumoto, 2003).

<sup>8</sup>If the (deprecated) Conversion 1 is used, the new features improve the UAS of *dpo3* from 93.04 to 93.51.

Model	UAS	Exact Match	Conjunctions	Prepositions
MSTParser (proj)	91.96	38.9	84.0	84.2
MSTParser (non-proj)	91.98	38.7	83.8	84.6
Berkeley (converted)	90.98	36.0	85.6	84.3
dpo3 (Grand+Sib)	93.21	<b>44.8</b>	<b>89.6</b>	<b>86.9</b>
dpo3+Unlabeled (Edges)	93.12	43.6	85.3	<b>87.0</b>
dpo3+Unlabeled (Sib)	93.15	43.7	85.5	86.8
dpo3+Unlabeled (Grand)	<b>93.55</b>	<b>46.1</b>	<b>90.6</b>	<b>87.5</b>
dpo3+Unlabeled (Grand+Sib)	<b>93.54</b>	<b>46.0</b>	<b>90.8</b>	<b>87.4</b>
- Clusters	93.10	<b>45.0</b>	<b>90.5</b>	<b>87.5</b>
- Prep,Conj Counts	<b>93.52</b>	<b>45.8</b>	<b>89.9</b>	<b>87.1</b>

Table 1: Test set accuracies under Conversion 2 of unlabeled attachment scores, complete sentence exact match accuracies, conjunction accuracy, and preposition accuracy. Bolded items are the best in each column, or not significantly different from the best in that column (sign test,  $p < .05$ ).

## 6.2 Impact of Unlabeled Data

The unlabeled data features improved the already state-of-the-art *dpo3* parser in UAS, complete sentence accuracy, conjunctions, and prepositions. However, because the sample sizes are much smaller for the latter three cases, only the UAS improvement is statistically significant.<sup>9</sup> Overall, the results in Table 1 show that while the inclusion of unlabeled data improves parser performance, increasing the size of factorization matters even more. Ablation experiments showed that cluster features have a larger impact on overall UAS, while count features have a larger impact on prepositions and conjunctions.

## 6.3 Comparison with Other Parsers

The resulting *dpo3*+Unlabeled parser is significantly better than both versions of MSTParser and the Berkeley parser converted to dependencies across all four evaluations. *dpo3*+Unlabeled has an UAS 1.5% higher than MSTParser, which has an UAS 1.0% higher than the converted constituency parser. The MSTParser uses sibling scoring, so it is unsurprising that it performs less well on the new conversion.

While the converted constituency parser is not as good on dependencies as MSTParser overall, note that it is over a percent and a half better than MSTParser on attaching conjunctions (85.6% versus 84.0%). Conjunction scope may benefit from parallelism and higher-level structure, which is easily accessible when joining two matching non-terminals

<sup>9</sup>There are 52,308 non-punctuation tokens in the test set, compared with 2416 sentences, 1373 conjunctions, and 5854 prepositions.

in a context-free grammar, but much harder to determine in the local views of graph-based dependency parsers. The dependencies arising from the Berkeley constituency trees have higher conjunction accuracies than either the edge-based or sibling-based *dpo3*+Unlabeled parser. However, once grandparents are included in the factorization, the *dpo3*+Unlabeled is significantly better at attaching conjunctions than the constituency parser, attaching conjunctions with an accuracy over 90%. Therefore, some of the disadvantages of dependency parsing compared with constituency parsing can be compensated for with larger factorizations.

Scoring	Conjunctions	
	Conversion 1 (deprecated)	Conversion 2
Edge	86.3	85.3
Sib	<b>87.8</b>	85.5
Grand	<b>87.2</b>	<b>90.6</b>
Grand+Sib	<b>88.3</b>	<b>90.8</b>

Table 2: Unlabeled attachment accuracy for conjunctions. Bolded items are the best in each column, or not significantly different (sign test,  $p < .05$ ).

## 6.4 Impact of Data Representation

Tables 2 and 3 show the results of the *dpo3*+Unlabeled parser for conjunctions and prepositions, respectively, under the two different conversions. The data representation has an impact on which factorizations perform best. Under Conversion 1, conjunctions are more accurate under a sibling parser than a grandparent parser, while the

Scoring	Prepositions	
	Conversion 1 (deprecated)	Conversion 2
Edge	87.4	<b>87.0</b>
Sib	87.5	86.8
Grand	<b>87.9</b>	<b>87.5</b>
Grand+Sib	<b>88.4</b>	<b>87.4</b>

Table 3: Unlabeled attachment accuracy for prepositions. Bolded items are the best in each column, or not significantly different (sign test,  $p < .05$ ).

pattern is reversed for Conversion 2.

Conjunctions show a much stronger need for higher order factorizations than prepositions do. This is not too surprising, as prepositions have more of a selectional preference than conjunctions, and so the preposition itself is more informative about where it should attach. While prepositions do improve with larger factorizations, the improvement beyond edge-based is not significant for Conversion 2. One hypothesis for why Conversion 1 shows more of an improvement is that the wider scope leads to the semantic head being included; in Conversion 2, the semantic head is chosen as the parent of the preposition, so the wider scope is less necessary.

### 6.5 Preposition Error Analysis

Prepositions are *still* the largest source of errors in the dpo3+Unlabeled parser. We therefore analyze the errors made on the development set to determine whether the difficult remaining cases for parsers correspond to the Hindle and Rooth (1993) style PP-attachment classification task. In the PP-attachment classification task, the two choices for where the preposition attaches are the previous verb or the previous noun, and the preposition itself has a noun object. The ones that *do* attach to the preceding noun or verb (not necessarily the preceding word) and have a noun object (2323 prepositions) are attached by the dpo3+Unlabeled grandparent-scoring parser with 92.4% accuracy, while those that do not fit that categorization (1703 prepositions) have the correct parent only 82.7% of the time.

Local attachments are more accurate — prepositions are attached with 94.8% accuracy if the correct parent is the immediately preceding word (2364 cases) and only 79.1% accuracy if it is not (1662 cases). The preference is not necessarily for low

attachments though: the prepositions whose parent is not the preceding word are attached more accurately if the parent is the root word (usually corresponding to the main verb) of the sentence (90.8%, 587 cases) than if the parent is lower in the tree (72.7%, 1075 cases).

## 7 Conclusion

Features derived from unlabeled data (clusters and web counts) significantly improve a state-of-the-art dependency parser for English. We showed how well various factorizations are able to take advantage of these unlabeled data features, focusing our analysis on conjunctions and prepositions. Including grandparents in the factorization increases the accuracy of conjunctions over 5% absolute over edge-based or sibling-based scoring. The representation of the data is extremely important for how the problem should be factored—under the old Penn2Malt dependency representation, a sibling parser was more accurate than a grandparent parser. As some important relationships were represented as siblings and some as grandparents, there was a need to develop third-order parsers which could exploit both simultaneously (Koo and Collins, 2010). Under the new pennconverter standard, a grandparent parser is significantly better than a sibling parser, and there is no significant improvement when including both.

## Acknowledgments

I would like to thank Terry Koo for making the *dpo3* parser publically available and for his help with using the parser. I would also like to thank Mitch Marcus and Kenneth Church for useful discussions. This material is based upon work supported under a National Science Foundation Graduate Research Fellowship.

## References

- M. Bansal and D. Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of ACL*, pages 693–702.
- S. Bergsma, D. Yarowsky, and K. Church. 2011. Using large monolingual and bilingual corpora to improve coordination disambiguation. In *Proceedings of ACL*, pages 1346–1355.

- P.F. Brown, P.V. Desouza, R.L. Mercer, V.J.D. Pietra, and J.C. Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.
- K.W. Church and P. Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.
- M. Collins, A. Globerson, T. Koo, X. Carreras, and P.L. Bartlett. 2008. Exponentiated gradient algorithms for conditional random fields and max-margin markov networks. *The Journal of Machine Learning Research*, 9:1775–1822.
- Michael Collins. 2002. Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*, pages 1–8.
- D. Gildea. 2001. Corpus variation and parser performance. In *Proceedings of EMNLP*, pages 167–202.
- D. Hindle and M. Rooth. 1993. Structural ambiguity and lexical relations. *Computational Linguistics*, 19(1):103–120.
- L. Huang and K. Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of ACL*, pages 1077–1086.
- X. Huang. 1983. Dealing with conjunctions in a machine translation environment. In *Proceedings of EACL*, pages 81–85.
- R. Johansson and P. Nugues. 2007. Extended constituent-to-dependency conversion for English. In *Proc. of the 16th Nordic Conference on Computational Linguistics (NODALIDA)*, pages 105–112.
- D. Jurafsky and J.H. Martin. 2008. *Speech and language processing: an introduction to natural language processing, computational linguistics and speech recognition*. Prentice Hall.
- T. Koo and M. Collins. 2010. Efficient third-order dependency parsers. In *Proceedings of ACL*, pages 1–11.
- T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL*, pages 595–603.
- T. Koo, A.M. Rush, M. Collins, T. Jaakkola, and D. Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of EMNLP*, pages 1288–1298.
- D. Lin, K. Church, H. Ji, S. Sekine, D. Yarowsky, S. Bergsma, K. Patil, E. Pitler, R. Lathbury, V. Rao, et al. 2010. New tools for web-scale n-grams. In *Proceedings of LREC*.
- 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.
- R. McDonald and F. Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL*, pages 81–88.
- R. McDonald, K. Crammer, and F. Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of ACL*, pages 91–98.
- J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. 2007. Malt-parser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.
- Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL*, pages 404–411.
- E. Pitler, S. Bergsma, D. Lin, and K. Church. 2010. Using web-scale n-grams to improve base np parsing performance. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 886–894.
- A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP*, pages 133–142.
- L. Schwartz, T. Aikawa, and C. Quirk. 2003. Disambiguation of English PP attachment using multilingual aligned data. In *Proceedings of MT Summit IX*.
- D. Vadas and J. Curran. 2007. Adding noun phrase structure to the Penn Treebank. In *ACL*, pages 240–247.
- R. Weischedel and A. Brunstein. 2005. BBN pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia*.
- R. Weischedel, M. Palmer, M. Marcus, E. Hovy, S. Pradhan, L. Ramshaw, N. Xue, A. Taylor, J. Kaufman, M. Franchini, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.
- H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of International Workshop of Parsing Technologies*, pages 195–206.
- Y. Zhang and S. Clark. 2008. A tale of two parsers: investigating and combining graph-based and transition-based dependency parsing using beam-search. In *Proceedings of EMNLP*, pages 562–571.
- G. Zhou, J. Zhao, K. Liu, and L. Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of ACL*, pages 1556–1565.

# Semi-supervised Dependency Parsing using Lexical Affinities

Seyed Abolghasem Mirroshandel<sup>†,\*</sup> Alexis Nasr<sup>†</sup> Joseph Le Roux<sup>◇</sup>

<sup>†</sup>Laboratoire d'Informatique Fondamentale de Marseille- CNRS - UMR 7279  
Université Aix-Marseille, Marseille, France

<sup>◇</sup>LIPN, Université Paris Nord & CNRS, Villetaneuse, France

<sup>\*</sup>Computer Engineering Department, Sharif university of Technology, Tehran, Iran

(ghasem.mirroshandel@lif.univ-mrs.fr, alexis.nasr@lif.univ-mrs.fr,  
leroux@univ-paris13.fr)

## Abstract

Treebanks are not large enough to reliably model precise lexical phenomena. This deficiency provokes attachment errors in the parsers trained on such data. We propose in this paper to compute lexical affinities, on large corpora, for specific lexico-syntactic configurations that are hard to disambiguate and introduce the new information in a parser. Experiments on the French Treebank showed a relative decrease of the error rate of 7.1% Labeled Accuracy Score yielding the best parsing results on this treebank.

## 1 Introduction

Probabilistic parsers are usually trained on treebanks composed of few thousands sentences. While this amount of data seems reasonable for learning syntactic phenomena and, to some extent, very frequent lexical phenomena involving closed parts of speech (POS), it proves inadequate when modeling lexical dependencies between open POS, such as nouns, verbs and adjectives. This fact was first recognized by (Bikel, 2004) who showed that bilexical dependencies were barely used in Michael Collins' parser.

The work reported in this paper aims at a better modeling of such phenomena by using a raw corpus that is several orders of magnitude larger than the treebank used for training the parser. The raw corpus is first parsed and the computed *lexical affinities* between lemmas, in specific lexico-syntactic configurations, are then injected back in the parser. Two outcomes are expected from this procedure, the first

is, as mentioned above, a better modeling of bilexical dependencies and the second is a method to adapt a parser to new domains.

The paper is organized as follows. Section 2 reviews some work on the same topic and highlights their differences with ours. In section 3, we describe the parser that we use in our experiments and give a detailed description of the frequent attachment errors. Section 4 describes how lexical affinities between lemmas are calculated and their impact is then evaluated with respect to the attachment errors made by the parser. Section 5 describes three ways to integrate the lexical affinities in the parser and reports the results obtained with the three methods.

## 2 Previous Work

Coping with lexical sparsity of treebanks using raw corpora has been an active direction of research for many years.

One simple and effective way to tackle this problem is to put together words that share, in a large raw corpus, similar linear contexts, into word *clusters*. The word occurrences of the training treebank are then replaced by their cluster identifier and a new parser is trained on the transformed treebank. Using such techniques (Koo et al., 2008) report significant improvement on the Penn Treebank (Marcus et al., 1993) and so do (Candito and Seddah, 2010; Candito and Crabbé, 2009) on the French Treebank (Abeillé et al., 2003).

Another series of papers (Volk, 2001; Nakov and Hearst, 2005; Pitler et al., 2010; Zhou et al., 2011) directly model word co-occurrences. Co-occurrences of pairs of words are first collected in a

raw corpus or internet  $n$ -grams. Based on the counts produced, lexical affinity scores are computed. The detection of pairs of words co-occurrences is generally very simple, it is either based on the direct adjacency of the words in the string or their co-occurrence in a window of a few words. (Bansal and Klein, 2011; Nakov and Hearst, 2005) rely on the same sort of techniques but use more sophisticated patterns, based on simple paraphrase rules, for identifying co-occurrences.

Our work departs from those approaches by the fact that we do not extract the lexical information directly on a raw corpus, but we first parse it and then extract the co-occurrences on the parse trees, based on some predetermined lexico-syntactic patterns. The first reason for this choice is that the linguistic phenomena that we are interested in, such as as PP attachment, coordination, verb subject and object can range over long distances, beyond what is generally taken into account when working on limited windows. The second reason for this choice was to show that the performances that the NLP community has reached on parsing, combined with the use of confidence measures allow to use parsers to extract accurate lexico-syntactic information, beyond what can be found in limited annotated corpora.

Our work can also be compared with self training approaches to parsing (McClosky et al., 2006; Suzuki et al., 2009; Steedman et al., 2003; Sagae and Tsujii, 2007) where a parser is first trained on a treebank and then used to parse a large raw corpus. The parses produced are then added to the initial treebank and a new parser is trained. The main difference between these approaches and ours is that we do not directly add the output of the parser to the training corpus, but extract precise lexical information that is then re-injected in the parser. In the self training approach, (Chen et al., 2009) is quite close to our work: instead of adding new parses to the treebank, the occurrence of simple interesting subtrees are detected in the parses and introduced as new features in the parser.

The way we introduce lexical affinity measures in the parser, in 5.1, shares some ideas with (Anguiano and Candito, 2011), who modify some attachments in the parser output, based on lexical information. The main difference is that we only take attachments that appear in an  $n$ -best parse list into account, while

they consider the first best parse and compute all potential alternative attachments, that may not actually occur in the  $n$ -best forests.

### 3 The Parser

The parser used in this work is the second order graph based parser (McDonald et al., 2005; Kübler et al., 2009) implementation of (Bohnet, 2010). The parser was trained on the French Treebank (Abeillé et al., 2003) which was transformed into dependency trees by (Candito et al., 2009). The size of the treebank and its decomposition into train, development and test sets is represented in table 1.

	nb of sentences	nb of words
FTB_TRAIN	9 881	278 083
FTB_DEV	1 239	36 508
FTB_TEST	1 235	36 340

Table 1: Size and decomposition of the French Treebank

The part of speech tagging was performed with the MELT tagger (Denis and Sagot, 2010) and lemmatized with the MACAON tool suite (Nasr et al., 2011). The parser gave state of the art results for parsing of French, reported in table 2.

	pred. POS tags		gold POS tags	
	punct	no punct	punct	no punct
LAS	88.02	90.24	88.88	91.12
UAS	90.02	92.50	90.71	93.20

Table 2: Labeled and unlabeled accuracy score for automatically predicted and gold POS tags with and without taking into account punctuation on FTB\_TEST.

Figure 1 shows the distribution of the 100 most common error types made by the parser. In this figure, x axis shows the error types and y axis shows the error ratio of the related error type ( $\frac{\text{number of errors of the specific type}}{\text{total number of errors}}$ ). We define an error type by the POS tag of the governor and the POS tag of the dependent. The figure presents a typical Zipfian distribution with a low number of frequent error types and a large number of unfrequent error types. The shape of the curve shows that concentrating on some specific frequent errors in order to increase the parser accuracy is a good strategy.

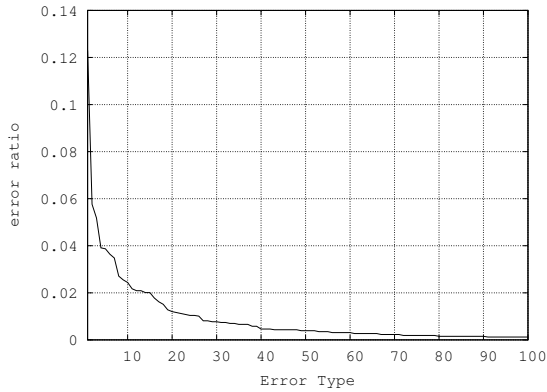


Figure 1: Distribution of the types of errors

Table 3 gives a finer description of the most common types of error made by the parser. Here we define more precise patterns for errors, where some lexical values are specified (for prepositions) and, in some cases, the nature of the dependency is taken into account. Every line of the table corresponds to one type of error. The first column describes the error type. The second column indicates the frequency of this type of dependency in the corpus. The third one displays the accuracy for this type of dependency (the number of dependencies of this type correctly analyzed by the parser divided by the total number of dependencies of this type). The fourth column shows the contribution of the errors made on this type of dependency to the global error rate. The last column associates a name with some of the error types that will prove useful in the remainder of the paper to refer to the error type.

Table 3 shows two different kinds of errors that impact the global error rate. The first one concerns very common dependencies that have a high accuracy but, due to their frequency, hurt the global error rate of the parser. The second one concerns low frequency, low accuracy dependency types. Lines 2 and 3, respectively attachment of the preposition  $\grave{a}$  to a verb and the subject dependency illustrate such a contrast. They both impact the total error rate in the same way (2.53% of the errors). But the first one is a low frequency low accuracy type (respectively 0.88% and 69.11%) while the second is a high frequency high accuracy type (respectively 3.43% and 93.03%). We will see in 4.2.2 that our method behaves quite differently on these two types of error.

dependency	freq.	acc.	contrib.	name
N→N	1.50	72.23	2.91	
V → à	0.88	69.11	2.53	VaN
V—subj → N	3.43	93.03	2.53	SBJ
N → CC	0.77	69.78	2.05	NcN
N → de	3.70	92.07	2.05	NdeN
V → de	0.66	74.68	1.62	VdeN
V—obj → N	2.74	90.43	1.60	OBJ
V → en	0.66	81.20	1.24	
V → pour	0.46	67.78	1.10	
N → ADJ	6.18	96.60	0.96	ADJ
N → à	0.29	70.64	0.72	NaN
N → pour	0.12	38.64	0.67	
N → en	0.15	47.69	0.57	

Table 3: The 13 most common error types

## 4 Creating the Lexical Resource

The lexical resource is a collection of tuples  $\langle C, g, d, s \rangle$  where  $C$  is a lexico-syntactic configuration,  $g$  is a lemma, called the governor of the configuration,  $d$  is another lemma called the dependent and  $s$  is a numerical value between 0 and 1, called the lexical affinity score, which accounts for the strength of the association between  $g$  and  $d$  in the context  $C$ . For example the tuple  $\langle (V, g) \xrightarrow{obj} (N, d), eat, oyster, 0.23 \rangle$  defines a simple configuration  $(V, g) \xrightarrow{obj} (N, d)$  that is an object dependency between verb  $g$  and noun  $d$ . When replacing variables  $g$  and  $d$  in  $C$  respectively with  $eat$  and  $oyster$ , we obtain the fully specified lexico syntactic pattern  $(V, eat) \xrightarrow{obj} (N, oyster)$ , that we call an instantiated configuration. The numerical value 0.23 accounts for how much  $eat$  and  $oyster$  like to co-occur in the verb-object configuration. Configurations can be of arbitrary complexity but they have to be generic enough in order to occur frequently in a corpus yet be specific enough to model a precise lexico syntactic phenomenon. The context  $(*, g) \xrightarrow{*} (*, d)$ , for example is very generic but does not model a precise linguistic phenomenon, as selectional preferences of a verb, for example. Moreover, configurations need to be error-prone. In the perspective of increasing a parser performances, there is no point in computing lexical affinity scores between words that appear in a configuration for which

the parser never makes mistakes.

The creation of the lexical resource is a three stage process. The first step is the definition of configurations, the second one is the collection of raw counts from the machine parsed corpora and the third one is the computation of lexical affinities based on the raw counts. The three steps are described in the following subsection while the evaluation of the created resource is reported in subsection 4.2.

#### 4.1 Computing Lexical Affinities

A set of 9 configurations have been defined. Their selection is a manual process based on the analysis of the errors made by the parser, described in section 3, as well as on the linguistic phenomena they model. The list of the 9 configurations is described in Table 4. As one can see on this table, configurations are usually simple, made up of one or two dependencies. Linguistically, configurations OBJ and SBJ concern subject and object attachments, configuration ADJ is related to attachments of adjectives to nouns and configurations NdeN, VdeN, VaN, and NaN indicate prepositional attachments. We have restricted ourselves here to two common French prepositions *à* and *de*. Configurations NcN and VcV deal respectively with noun and verb coordination.

Name	Description
OBJ	$(V, g) \xrightarrow{obj} (N, d)$
SBJ	$(V, g) \xrightarrow{subj} (N, d)$
ADJ	$(N, g) \rightarrow ADJ$
NdeN	$(N, g) \rightarrow (P, de) \rightarrow (N, d)$
VdeN	$(V, g) \rightarrow (P, de) \rightarrow (N, d)$
NaN	$(N, g) \rightarrow (P, \grave{a}) \rightarrow (N, d)$
VaN	$(V, g) \rightarrow (P, \grave{a}) \rightarrow (N, d)$
NcN	$(N, g) \rightarrow (CC, *) \rightarrow (N, d)$
VcV	$(V, g) \rightarrow (CC, *) \rightarrow (V, d)$

Table 4: List of the 9 configurations.

The computation of the number of occurrences of an instantiated configuration in the corpus is quite straightforward, it consists in traversing the dependency trees produced by the parser and detect the occurrences of this configuration.

At the end of the counts collection, we have gath-

CORPUS	Sent. nb.	Tokens nb.
AFP	1 024 797	31 486 618
EST REP	1 103 630	19 635 985
WIKI	1 592 035	33 821 460
TOTAL	3 720 462	84 944 063

Table 5: sizes of the corpora used to gather lexical counts

ered for every lemma  $l$  its number of occurrences as governor (resp. dependent) of configuration  $C$  in the corpus, noted  $\mathcal{C}(C, l, *)$  (resp.  $\mathcal{C}(C, *, l)$ ), as well as the number of occurrences of configuration  $C$  with lemma  $l_g$  as a governor and lemma  $l_d$  as a dependent, noted  $\mathcal{C}(C, l_g, l_d)$ . We are now in a position to compute the score  $s(C, l_g, l_d)$ . This score should reflect the tendency of  $l_g$  and  $l_d$  to appear together in configuration  $C$ . It should be maximal if whenever  $l_g$  occurs as the governor of configuration  $C$ , the dependent position is occupied by  $l_d$  and, symmetrically, if whenever  $l_d$  occurs as the dependent of configuration  $C$ , the governor position is occupied by  $l_g$ . A function that conforms such a behavior is the following:

$$s(C, l_g, l_d) = \frac{1}{2} \left( \frac{\mathcal{C}(C, l_g, l_d)}{\mathcal{C}(C, l_g, *)} + \frac{\mathcal{C}(C, l_g, l_d)}{\mathcal{C}(C, *, l_d)} \right)$$

it takes its values between 0 ( $l_g$  and  $l_d$  never co-occur) and 1 ( $g$  and  $d$  always co-occur). This function is close to pointwise mutual information (Church and Hanks, 1990) but takes its values between 0 and 1.

#### 4.2 Evaluation

Lexical affinities were computed on three corpora of slightly different genres. The first one, is a collection of news report of the French press agency *Agence France Presse*, the second is a collection of newspaper articles from a local French newspaper : *l'Est Républicain*. The third one is a collection of articles from the French Wikipedia. The size of the different corpora are detailed in table 5. The corpus was first POS tagged, lemmatized and parsed in order to get the 50 best parses for every sentence. Then the lexical resource was built, based on the 9 configurations described in table 4.

The lexical resource has been evaluated on FTB\_DEV with respect to two measures: coverage



and correction rate, described in the next two sections.

#### 4.2.1 Coverage

Coverage measures the instantiated configurations present in the evaluation corpus that are in the resource. The results are presented in table 6. Every line represents a configuration, the second column indicates the number of different instantiations of this configuration in the evaluation corpus, the third one indicates the number of instantiated configurations that were actually found in the lexical resource and the fourth column shows the coverage for this configuration, which is the ratio third column over the second. Last column represents the coverage of the training corpus (the lexical resource is extracted on the training corpus) and the last line represents the same quantities computed on all configurations.

Table 6 shows two interesting results: firstly the high variability of coverage with respect to configurations, and secondly the low coverage when the lexical resource is computed on the training corpus, this fact being consistent with the conclusions of (Bikel, 2004). A parser trained on a treebank cannot be expected to reliably select the correct governor in lexically sensitive cases.

Conf.	occ.	pres.	cov.	T cov.
OBJ	1017	709	0.70	0.21
SBJ	1210	825	0.68	0.24
ADJ	1791	1239	0.69	0.33
NdeN	1909	1287	0.67	0.31
VdeN	189	107	0.57	0.16
NaN	123	61	0.50	0.20
VaN	422	273	0.65	0.23
NcN	220	55	0.25	0.10
VcV	165	93	0.56	0.04
$\Sigma$	7046	4649	0.66	0.27

Table 6: Coverage of the lexical resource over FTB\_DEV.

#### 4.2.2 Correction Rate

While coverage measures how many instantiated configurations that occur in the treebank are actually present in the lexical resource, it does not measure if the information present in the lexical resource can actually help correcting the errors made by the parser.

We define *Correction Rate* (CR) as a way to approximate the usefulness of the data. Given a word  $d$  present in a sentence  $S$  and a configuration  $C$ , the set of all potential governors of  $d$  in configuration  $C$ , in all the  $n$ -best parses produced by the parser is computed. This set is noted  $\mathcal{G} = \{g_1, \dots, g_j\}$ . Let us note  $G_L$  the element of  $\mathcal{G}$  that maximizes the lexical affinity score. When the lexical resource gives no score to any of the elements of  $\mathcal{G}$ ,  $G_L$  is left unspecified.

Ideally,  $\mathcal{G}$  should not be the set of governors in the  $n$ -best parses but the set of all possible governors for  $d$  in sentence  $S$ . Since we have no simple way to compute the latter, we will content ourselves with the former as an approximation of the latter.

Let us note  $G_H$  the governor of  $d$  in the (first) best parse produced and  $G_R$  the governor of  $d$  in the correct parse. CR measures the effect of replacing  $G_H$  with  $G_L$ .

We have represented in table 7 the different scenarios that can happen when comparing  $G_H$ ,  $G_R$  and  $G_L$ .

$G_H = G_R$	$G_L = G_R$ or $G_L$ unspec.	CC
	$G_L \neq G_R$	CE
$G_H \neq G_R$	$G_L = G_R$	EC
	$G_L \neq G_R$ or $G_L$ unspec.	EE
	$G_R \notin \mathcal{G}$	NA

Table 7: Five possible scenarios when comparing the governor of a word produced by the parser ( $G_H$ ), in the reference parse ( $G_R$ ) and according to the lexical resource ( $G_L$ ).

In scenarios CC and CE, the parser did not make a mistake (the first letter, C, stands for correct). In scenario CC, the lexical affinity score was compatible with the choice of the parser or the lexical resource did not select any candidate. In scenario CE, the lexical resource introduced an error. In scenarios EC and EE, the parser made an error. In EC, the error was corrected by the lexical resource while in EE, it wasn't. Either because the lexical resource candidate was not the correct governor or it was unspecified. The last case, NA, indicates that the correct governor does not appear in any of the  $n$ -best parses. Technically this case could be integrated in EE (an error made by the parser was not corrected by the lexical resource) but we chose to keep it apart

since it represents a case where the right solution could not be found in the  $n$ -best parse list (the correct governor is not a member of set  $\mathcal{G}$ ).

Let's note  $n_S$  the number of occurrences of scenario  $S$  for a given configuration. We compute CR for this configuration in the following way:

$$\begin{aligned} CR &= \frac{\text{old error number} - \text{new error number}}{\text{old error number}} \\ &= \frac{n_{EC} - n_{CE}}{n_{EE} + n_{EC} + n_{NA}} \end{aligned}$$

When CR is equal to 0, the correction did not have any impact on the error rate. When  $CR > 0$ , the error rate is reduced and if  $CR < 0$  it is increased<sup>1</sup>.

CR for each configuration is reported in table 8. The counts of the different scenarios have also been reported.

Conf.	$n_{CC}$	$n_{CE}$	$n_{EC}$	$n_{EE}$	$n_{NA}$	CR
OBJ	992	30	51	5	17	0.29
SBJ	1131	35	61	16	34	0.23
ADJ	2220	42	16	20	6	-0.62
NdeN	2083	93	42	44	21	-0.48
VdeN	150	2	49	1	13	0.75
NaN	89	5	21	10	2	0.48
VaN	273	19	132	8	11	0.75
NcN	165	17	12	31	12	-0.09
VcN	120	21	14	11	5	-0.23
$\Sigma$	7223	264	398	146	121	0.20

Table 8: Correction Rate of the lexical resource with respect to FTB.DEV.

Table 8 shows very different results among configurations. Results for PP attachments VdeN, VaN and NaN are quite good (a CR of 75% for a given configuration, as VdeN indicates that the number of errors on such a configuration is decreased by 25%). It is interesting to note that the parser behaves quite badly on these attachments: their accuracy (as reported in table 3) is, respectively 74.68, 69.1 and 70.64. Lexical affinity helps in such cases. On the other hand, some attachments like configuration ADJ and NdeN, for which the parser showed very good accuracy (96.6 and 92.2) show very poor performances. In such cases, taking into account lexical affinity creates new errors.

<sup>1</sup>One can note, that contrary to coverage, CR does not measure a characteristic of the lexical resource alone, but the lexical resource combined with a parser.

On average, using the lexical resource with this simple strategy of systematically replacing  $G_H$  with  $G_L$  allows to decrease by 20% the errors made on our 9 configurations and by 2.5% the global error rate of the parser.

### 4.3 Filtering Data with Ambiguity Threshold

The data used to extract counts is noisy: it contains errors made by the parser. Ideally, we would like to take into account only non ambiguous sentences, for which the parser outputs a single parse hypothesis, hopefully the good one. Such an approach is obviously doomed to fail since almost every sentence will be associated to several parses. Another solution would be to select sentences for which the parser has a high confidence, using confidence measures as proposed in (Sánchez-Sáez et al., 2009; Hwa, 2004). But since we are only interested in some parts of sentences (usually one attachment), we don't need high confidence for the whole sentence. We have instead used a parameter, defined on single dependencies, called the *ambiguity measure*.

Given the  $n$  best parses of a sentence and a dependency  $\delta$ , present in at least one of the  $n$  best parses, let us note  $\mathcal{C}(\delta)$  the number of occurrences of  $\delta$  in the  $n$  best parse set. We note  $AM(\delta)$  the ambiguity measure associated to  $\delta$ . It is computed as follows:

$$AM(\delta) = 1 - \frac{\mathcal{C}(\delta)}{n}$$

An ambiguity measure of 0 indicates that  $\delta$  is non ambiguous in the set of the  $n$  best parses (the word that constitutes the dependent in  $\delta$  is attached to the word that constitutes the governor in  $\delta$  in all the  $n$ -best analyses). When  $n$  gets large enough this measure approximates the non ambiguity of a dependency in a given sentence.

Ambiguity measure is used to filter the data when counting the number of occurrences of a configuration: only occurrences that are made of dependencies  $\delta$  such that  $AM(\delta) \leq \tau$  are taken into account.  $\tau$  is called the ambiguity threshold.

The results of coverage and CR given above were computed for  $\tau$  equal to 1, which means that, when collecting counts, all the dependencies are taken into account whatever their ambiguity is. Table 9 shows coverage and CR for different values of  $\tau$ . As expected, coverage decreases with  $\tau$ . But, interest-

ingly, decreasing  $\tau$ , from 1 down to 0.2 has a positive influence on CR. Ambiguity threshold plays the role we expected: it allows to reduce noise in the data, and corrects more errors.

	$\tau = 1.0$	$\tau = 0.4$	$\tau = 0.2$	$\tau = 0.0$
	cov/CR	cov/CR	cov/CR	cov/CR
OBJ	0.70/0.29	0.58/0.36	0.52/0.36	0.35/0.38
SBJ	0.68/0.23	0.64/0.23	0.62/0.23	0.52/0.23
ADJ	0.69/-0.62	0.61/-0.52	0.56/-0.52	0.43/-0.38
NdeN	0.67/-0.48	0.58/-0.53	0.52/-0.52	0.38/-0.41
VdeN	0.57/0.75	0.44/0.73	0.36/0.73	0.20/0.30
NaN	0.50/0.48	0.34/0.42	0.28/0.45	0.15/0.48
VaN	0.65/0.75	0.50/0.8	0.41/0.80	0.26/0.48
NcN	0.25/-0.09	0.19/0	0.16/0.02	0.07/0.13
VcV	0.56/-0.23	0.42/-0.07	0.28/0.03	0.08/0.07
Avg	0.66/0.2	0.57/0.23	0.51/0.24	0.38/0.17

Table 9: Coverage and Correction Rate on FTB.DEV for several values of ambiguity threshold.

## 5 Integrating Lexical Affinity in the Parser

We have devised three methods for taking into account lexical affinity scores in the parser. The first two are post-processing methods, that take as input the  $n$ -best parses produced by the parser and modify some attachments with respect to the information given by the lexical resource. The third method introduces the lexical affinity scores as new features in the parsing model. The three methods are described in 5.1, 5.2 and 5.3. They are evaluated in 5.4.

### 5.1 Post Processing Method

The post processing method is quite simple. It is very close to the method that was used to compute the Correction Rate of the lexical resource, in 4.2.2: it takes as input the  $n$ -best parses produced by the parser and, for every configuration occurrence  $C$  found in the first best parse, the set ( $\mathcal{G}$ ) of all potential governors of  $C$ , in the  $n$ -best parses, is computed and among them, the word that maximizes the lexical affinity score ( $G_L$ ) is identified.

Once  $G_L$  is identified, one can replace the choice of the parser ( $G_H$ ) with  $G_L$ . This method is quite crude since it does not take into account the confidence the parser has in the solution proposed. We observed, in 4.2.2 that CR was very low for configurations for which the parser achieves good accuracy. In order to introduce the parser confidence in the final choice of a governor, we compute  $\mathcal{C}(G_H)$  and

$\mathcal{C}(G_L)$  which respectively represent the number of times  $G_H$  and  $G_L$  appear as the governor of configuration  $C$ . The choice of the final governor, noted  $\hat{G}$ , depends on the ratio of  $\mathcal{C}(G_H)$  and  $\mathcal{C}(G_L)$ . The complete selection strategy is the following:

1. if  $G_H = G_L$  or  $G_L$  is unspecified,  $\hat{G} = G_H$ .
2. if  $G_H \neq G_L$ ,  $\hat{G}$  is determined as follows:

$$\hat{G} = \begin{cases} G_H & \text{if } \frac{\mathcal{C}(G_H)}{\mathcal{C}(G_L)} > \alpha \\ G_L & \text{otherwise} \end{cases}$$

where  $\alpha$  is a coefficient that is optimized on the development data set.

We have reported, in table 10 the values of CR, for the 9 different features, using this strategy, for  $\tau = 1$ . We do not report the values of CR for other values of  $\tau$  since they are very close to each other. The table shows several noticeable facts. First, the new strategy performs much better than the former one (crudely replacing  $G_H$  by  $G_L$ ), the value of CR increased from 0.2 to 0.4, which means that the errors made on the nine configurations are now decreased by 40%. Second, CR is now positive for every configuration: the number of errors is decreased for every configuration.

Conf.	OBJ	SUJ	ADJ	NdeN	VdeN
CR	0.45	0.46	0.14	0.05	0.73
Conf.	NaN	VaN	NcN	VcV	$\Sigma$
CR	0.12	0.8	0.12	0.1	0.4

Table 10: Correction Rate on FTB.DEV when taking into account parser confidence.

### 5.2 Double Parsing Method

The post processing method performs better than the naive strategy that was used in 4.2.2. But it has an important drawback: it creates inconsistent parses. Recall that the parser we are using is based on a second order model, which means that the score of a dependency depends on some neighboring ones. Since with the post processing method only a subset of the dependencies are modified, the resulting parse is inconsistent: the score of some dependencies is computed on the basis of other dependencies that have been modified.

In order to compute a new optimal parse tree that preserves the modified dependencies, we have used a technique proposed in (Mirroshandel and Nasr, 2011) that modifies the scoring function of the parser in such a way that the dependencies that we want to keep in the parser output get better scores than all competing dependencies.

The double parsing method is therefore a three stage method. First, sentence  $S$  is parsed, producing the  $n$ -best parses. Then, the post processing method is used, modifying the first best parse. Let's note  $\mathcal{D}$  the set of dependencies that were changed in this process. In the last stage, a new parse is produced, that preserves  $\mathcal{D}$ .

### 5.3 Feature Based Method

In the feature based method, new features are added to the parser that rely on lexical affinity scores. These features are of the following form:  $\langle C, l_g, l_d, \delta_C(s) \rangle$ , where  $C$  is a configuration number,  $s$  is the lexical affinity score ( $s = s(C, l_g, l_d)$ ) and  $\delta_C(\cdot)$  is a discretization function.

Discretization of the lexical affinity scores is necessary in order to fight against data sparseness. In this work, we have used Weka software (Hall et al., 2009) to discretize the scores with unsupervised binning. Binning is a simple process which divides the range of possible values a parameter can take into subranges called bins. Two methods are implemented in Weka to find the optimal number of bins: equal-frequency and equal-width. In equal-frequency binning, the range of possible values are divided into  $k$  bins, each of which holds the same number of instances. In equal-width binning, which is the method we have used, the range are divided into  $k$  subranges of the same size. The optimal number of bins is the one that minimizes the entropy of the data. Weka computes different number of bins for different configurations, ranging from 4 to 10. The number of new features added to the parser is equal to  $\sum_C B(C)$  where  $C$  is a configuration and  $B(C)$  is the number of bins for configuration  $C$ .

### 5.4 Evaluation

The three methods described above have been evaluated on FTB\_TEST. Results are reported in table 11. The three methods outperformed the baseline (the state of the art parser for French which is a second

order graph based method) (Bohnet, 2010). The best performances were obtained by the Double Parsing method that achieved a labeled relative error reduction of 7,1% on predicted POS tags, yielding the best parsing results on the French Treebank. It performs better than the Post Processing method, which means that the second parsing stage corrects some inconsistencies introduced in the Post Processing method. The performances of the Feature Based method are disappointing, it achieves an error reduction of 1.4%. This result is not easy to interpret. It is probably due to the limited number of new features introduced in the parser. These new features probably have a hard time competing with the large number of other features in the training process.

		pred. POS tags		gold POS tags	
		punct	no punct	punct	no punct
BL	LAS	88.02	90.24	88.88	91.12
	UAS	90.02	92.50	90.71	93.20
PP	LAS	88.45	90.73	89.46	91.78
	UAS	90.61	93.20	91.44	93.86
DP	LAS	<b>88.87</b>	<b>91.10</b>	<b>89.72</b>	<b>91.90</b>
	UAS	<b>90.84</b>	<b>93.30</b>	<b>91.58</b>	<b>93.99</b>
FB	LAS	88.19	90.33	89.29	91.43
	UAS	90.22	92.62	91.09	93.46

Table 11: Parser accuracy on FTB\_TEST using the standard parser (BL) the post processing method (PP), the double parsing method (DP) and the feature based method.

## 6 Conclusion

Computing lexical affinities, on large corpora, for specific lexico-syntactic configurations that are hard to disambiguate has shown to be an effective way to increase the performances of a parser. We have proposed in this paper one method to compute lexical affinity scores as well as three ways to introduce this new information in a parser. Experiments on a French corpus showed a relative decrease of the error rate of 7.1% Labeled Accuracy Score.

## Acknowledgments

This work has been funded by the French Agence Nationale pour la Recherche, through the projects SEQUOIA (ANR-08-EMER-013) and EDYLEX (ANR-08-CORD-009).

## References

- A. Abeillé, L. Clément, and F. Toussanel. 2003. Building a treebank for french. In Anne Abeillé, editor, *Treebanks*. Kluwer, Dordrecht.
- E.H. Anguiano and M. Candito. 2011. Parse correction with specialized models for difficult attachment types. In *Proceedings of EMNLP*.
- M. Bansal and D. Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of ACL*, pages 693–702.
- D. Bikel. 2004. Intricacies of Collins’ parsing model. *Computational Linguistics*, 30(4):479–511.
- B. Bohnet. 2010. Very high accuracy and fast dependency parsing is not a contradiction. In *Proceedings of ACL*, pages 89–97.
- M. Candito and B. Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. In *Proceedings of the 11th International Conference on Parsing Technologies*, pages 138–141.
- M. Candito and D. Seddah. 2010. Parsing word clusters. In *Proceedings of the NAACL HLT Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 76–84.
- M. Candito, B. Crabbé, P. Denis, and F. Guérin. 2009. Analyse syntaxique du français : des constituants aux dépendances. In *Proceedings of Traitement Automatique des Langues Naturelles*.
- W. Chen, J. Kazama, K. Uchimoto, and K. Torisawa. 2009. Improving dependency parsing with subtrees from auto-parsed data. In *Proceedings of EMNLP*, pages 570–579.
- K.W. Church and P. Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.
- P. Denis and B. Sagot. 2010. Exploitation d’une ressource lexicale pour la construction d’un étiqueteur morphosyntaxique état-de-l’art du français. In *Proceedings of Traitement Automatique des Langues Naturelles*.
- M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. 2009. The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.
- R. Hwa. 2004. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276.
- T. Koo, X. Carreras, and M. Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of the ACL HLT*, pages 595–603.
- S. Kübler, R. McDonald, and J. Nivre. 2009. Dependency parsing. *Synthesis Lectures on Human Language Technologies*, 1(1):1–127.
- 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.
- D. McClosky, E. Charniak, and M. Johnson. 2006. Effective self-training for parsing. In *Proceedings of HLT NAACL*, pages 152–159.
- R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of HLT-EMNLP*, pages 523–530.
- S.A. Mirroshandel and A. Nasr. 2011. Active learning for dependency parsing using partially annotated sentences. In *Proceedings of International Conference on Parsing Technologies*.
- P. Nakov and M. Hearst. 2005. Using the web as an implicit training set: application to structural ambiguity resolution. In *Proceedings of HLT-EMNLP*, pages 835–842.
- A. Nasr, F. Béchet, J-F. Rey, B. Favre, and Le Roux J. 2011. MACAON: An NLP tool suite for processing word lattices. In *Proceedings of ACL*.
- E. Pitler, S. Bergsma, D. Lin, and K. Church. 2010. Using web-scale N-grams to improve base NP parsing performance. In *Proceedings of COLING*, pages 886–894.
- K. Sagae and J. Tsujii. 2007. Dependency parsing and domain adaptation with lr models and parser ensembles. In *Proceedings of the CoNLL shared task session of EMNLP-CoNLL*, volume 7, pages 1044–1050.
- R. Sánchez-Sáez, J.A. Sánchez, and J.M. Benedí. 2009. Statistical confidence measures for probabilistic parsing. In *Proceedings of RANLP*, pages 388–392.
- M. Steedman, M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker, and J. Crim. 2003. Bootstrapping statistical parsers from small datasets. In *Proceedings of EACL*, pages 331–338.
- J. Suzuki, H. Isozaki, X. Carreras, and M. Collins. 2009. An empirical study of semi-supervised structured conditional models for dependency parsing. In *Proceedings of EMNLP*, pages 551–560.
- M. Volk. 2001. Exploiting the WWW as a corpus to resolve PP attachment ambiguities. In *Proceedings of Corpus Linguistics*.
- G. Zhou, J. Zhao, K. Liu, and L. Cai. 2011. Exploiting web-derived selectional preference to improve statistical dependency parsing. In *Proceedings of HLT-ACL*, pages 1556–1565.

# Chinese Comma Disambiguation for Discourse Analysis

**Yaqin Yang**  
Brandeis University  
415 South Street  
Waltham, MA 02453, USA  
yaqin@brandeis.edu

**Nianwen Xue**  
Brandeis University  
415 South Street  
Waltham, MA 02453, USA  
xuen@brandeis.edu

## Abstract

The Chinese comma signals the boundary of discourse units and also anchors discourse relations between adjacent text spans. In this work, we propose a discourse structure-oriented classification of the comma that can be automatically extracted from the Chinese Treebank based on syntactic patterns. We then experimented with two supervised learning methods that automatically disambiguate the Chinese comma based on this classification. The first method integrates comma classification into parsing, and the second method adopts a “post-processing” approach that extracts features from automatic parses to train a classifier. The experimental results show that the second approach compares favorably against the first approach.

## 1 Introduction

The Chinese comma, which looks graphically very similar to its English counterpart, is functionally quite different. It has attracted a significant amount of research that studied the problem from the viewpoint of natural language processing. For example, Jin et al (2004) and Li et al (2005) view the disambiguation of the Chinese comma as a way of breaking up long Chinese sentences into shorter ones to facilitate parsing. The idea is to split a long sentence into multiple comma-separated segments, parse them individually, and reconstruct the syntactic parse for the original sentence. Although both studies show a positive impact of this approach, comma disambiguation is viewed merely as a convenient tool to help achieve a more important goal.

Xue and Yang (2011) point out that the very reason for the existence of these long Chinese sentences is because the Chinese comma is ambiguous and in some context, it identifies the boundary of a sentence just as a period, a question mark, or an exclamation mark does. The disambiguation of comma is viewed as a necessary step to detect sentence boundaries in Chinese and it can benefit a whole range of downstream NLP applications such as syntactic parsing and Machine Translation. In Machine Translation, for example, it is very typical for “one” Chinese sentence to be translated into multiple English sentences, with each comma-separated segment corresponding to one English sentence. In the present work, we expand this view and propose to look at the Chinese comma in the context of discourse analysis. The Chinese comma is viewed as a delimiter of elementary discourse units (EDUs), in the sense of the Rhetorical Structure Theory (Carlson et al., 2002; Mann et al., 1988). It is also considered to be the anchor of discourse relations, in the sense of the Penn Discourse Treebank (PDT) (Prasad et al., 2008). Disambiguating the comma is thus necessary for the purpose of discourse segmentation, the identification of EDUs, a first step in building up the discourse structure of a Chinese text.

Developing a supervised or semi-supervised model of discourse segmentation would require ground truth annotated based on a well-established representation scheme, but as of right now no such annotation exists for Chinese to the best of our knowledge. However, syntactically annotated treebanks often contain important clues that can be used to infer discourse-level information. We present

a method of automatically deriving a preliminary form of discourse structure anchored by the Chinese comma from the Penn Chinese Treebank (CTB) (Xue et al., 2005), and using this information to train and test supervised models. This discourse information is formalized as a classification of the Chinese comma, with each class representing the boundary of an elementary discourse unit as well as the anchor of a coarse-grained discourse relation between the two discourse units that it delimits. We then develop two comma classification methods. In the first method, we replace the part-of-speech (POS) tag of each comma in the CTB with a derived discourse category and retrain a state-of-the-art Chinese parser on the relabeled data. We then evaluate how accurately the commas are classified in the parsing process. In the second method, we parse these sentences and extract lexical and syntactic information as features to predict these new discourse categories. The second approach gives us more control over what features to extract and our results show that it compares favorably against the first approach.

The rest of the paper is organized as follows. In Section 2, we present our approach to automatically extract discourse information from a syntactically annotated treebank and present our classification scheme. In Section 3, we describe our supervised learning methods and the features we extracted. Section 4 presents our experiment setup and experimental results. Related work is reviewed in Section 5. We conclude in Section 6.

## 2 Chinese comma classification

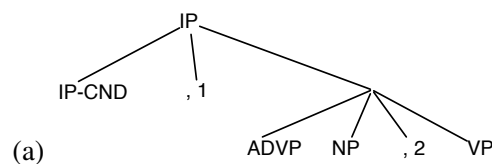
There are many ways to conceptualize the discourse structure of a text (Mann et al., 1988; Prasad et al., 2008), but there is more of a consensus among researchers about the fundamental building blocks of the discourse structure. For the Rhetorical Discourse Theory, the building blocks are Elementary Discourse Units (EDUs). For the PDT, the building blocks are abstract objects such as propositions, facts. Although they are phrased in different ways, syntactically these discourse units are generally realized as clauses or built on top of clauses. So the first step in building the discourse structure of a text is to identify these discourse units.

In Chinese, these elementary discourse units are generally delimited by the comma, but not all commas mark the boundaries of a discourse unit. In (1), for example, Comma [1] marks the boundary of a discourse unit while Comma [2] does not. This is reflected in its English translation: while the first comma corresponds to an English comma, the second comma is not translated at all, as it marks the boundary between a subject and its predicate, where no comma is needed in English. Disambiguating these two types of commas is thus an important first step in identifying elementary discourse units and building up the discourse structure of a text.

- (1) 王翔 虽 年 过 半 百, [1] 但  
Wang Xiang although age over 50 , but  
其 充 沛 的 精 力 和 敏 捷 的  
his abundant DE energy and quick DE  
思 维 , [2] 给 人 一 个 挑 战 者  
thinking , give people one CL challenger  
的 印 象 。  
DE impression .

“Although Wang Xiang is over 50 years old, his abundant energy and quick thinking leave people the impression of a challenger.”

Although to the best of our knowledge, no such discourse segmented data for Chinese exists in the public domain, this information can be extracted from the syntactic annotation of the CTB. In the syntactic annotation of the sentence, illustrated in (a), it is clear that while the first comma in the sentence marks the boundary of a clause, the second one marks the demarcation between the subject NP and the predicate VP and thus is not an indicator of a discourse boundary.



In addition to a binary distinction of whether a comma marks the boundary of a discourse unit, the CTB annotation also allows the extraction of a more elaborate classification of commas based on coordination and subordination relations of comma-separated clauses. This classification of the Chinese

comma can be viewed as a first approximation of the discourse relations anchored by the comma that can be refined later via a manual annotation process.

Based on the syntactic annotation in the CTB, we classify the Chinese comma into seven hierarchically organized categories, as illustrated in Figure 1. The first distinction is made between commas that indicate a discourse boundary (RELATION) and those that do not (OTHER). Commas that indicate discourse boundaries are further divided into commas that separate coordinated discourse units (COORD) vs commas that separate discourse units in a subordination relation (SUBORD). Based on the levels of embedding and the syntactic category of the coordinated structures, we define three different types of coordination (SB, IP\_COORD and VP\_COORD). We also define three types of subordination relations (ADJ, COMP, Sent\_SBJ), based on the syntactic structure. As we will show below, each of the six relations has a clear syntactic pattern that can be exploited for their automatic detection.

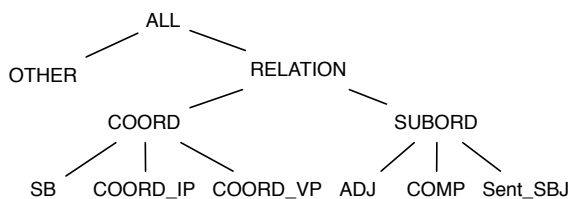


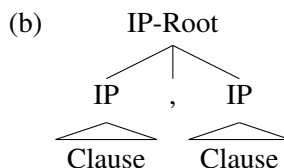
Figure 1: Comma classification

**Sentence Boundary (SB):** Following (Xue and Yang, 2011), we consider the loosely coordinated IPs that are the immediate children of the root IP to be independent sentences, and the commas separating them to be delimiters of sentence boundary. This is illustrated in (2), where a Chinese sentence can be split into two independent shorter sentences at the comma. We view this comma to be a marker of the sentence boundary and it serves the same function as the unambiguous sentence boundary delimiters (periods, question marks, exclamation marks) in Chinese. The syntactic pattern that is used to infer this relation is illustrated in (b).

- (2) 广东省 建立 了 自然  
Guangdong province establish ASP natural

科学 基金 , [3] 每年  
science foundation , every year  
投入 在一亿 元  
investment at one hundred million yuan  
以上 。  
above .

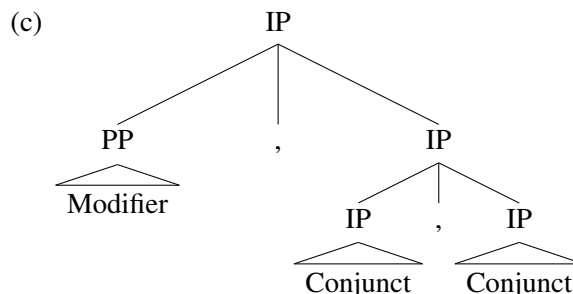
“Natural Science Foundation is established in Guangdong Province. More than one hundred million yuan is invested every year.”



**IP Coordination (IP\_COORD):** Coordinated IPs that are not the immediate children of the root IP are also considered to be discourse units and the commas linking them are labeled IP\_COORD. Different from the sentence boundary cases, these coordinated IPs are often embedded in a larger structure. An example is given in (3) and its typical syntactic pattern is illustrated in (c).

- (3) 据 陆仁法 介绍 , [4]  
According to Lu Renfa presentation ,  
全国 税收 任务已  
the whole country revenue goal already  
超额 完成 , [5] 总体  
exceeding quota complete , overall  
情况 比较 好 。  
situation fairly good .

“According to Lu Renfa, the national revenue goal is met and exceeded, and the overall situation is fairly good.”



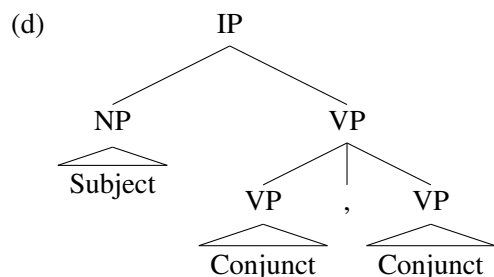
**VP Coordination (VP\_COORD):** Coordinated VPs, when separated by the comma, are not semantically different from coordinated IPs. The only difference is that in the latter case, the coordinated VPs



share a subject, while coordinated IPs tend to have different subjects. Maintaining this distinction allow us to model subject (dis)continuity, which helps recover a subject when it is dropped, a prevalent phenomenon in Chinese. As shown in (4), the VPs in the text spans separated by Comma [6] have the same subject, thus the subject in the second VP is dropped. The syntactic pattern that allows us to extract this structure is given in (d).

- (4) 中国 银行 是四大 国有  
 China Bank is four major state-owned  
 商业 银行之一 , [6] 也 是  
 commercial bank one of these , also is  
 中国 的 主要 外汇 银行 。  
 China DE major foreign exchange bank .

“Bank of China is one of the four major state-owned commercial banks, and it is also China’s major foreign exchange bank.”

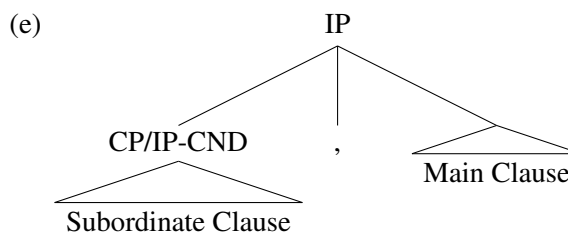


**Adjunction (ADJ):** Adjunction is one of three types of subordination relations we define. It holds between a subordinate clause and its main clause. The subordinate clause is normally introduced by a subordinating conjunction and it typically provides the cause, purpose, manner, or condition for the main clause. In the PDT terms, these subordinate conjunctions are discourse connectives that anchor a discourse relation between the subordinate clause and the main clause. In Chinese, with few exceptions, the subordinate clause comes before the main clause. (5) is an example of this relation.

- (5) 若工程 发生 保险 责任 范围  
 if project happen insurance liability scope  
 内 的 自然 灾害 , [7]  
 inside DE natural disaster ,  
 中保 财产 保险 公司  
 China Insurance property insurance company

将 按 规定 进行  
 will according to provision execute  
 赔偿 。  
 compensation .

“If natural disasters within the scope of the insurance liability happen in the project, PICC Property Insurance Company will provide compensations according to the provisions.”



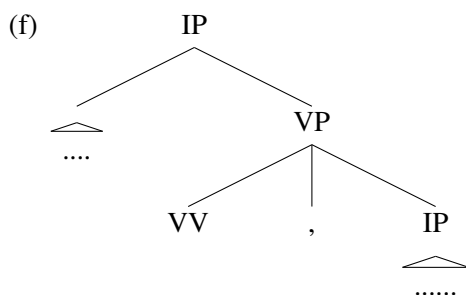
(e) shows how (5) is represented in the syntactic structure in the CTB. Extracting this relation requires more than just the syntactic configuration between these two clauses. We also take advantage of the functional (dash) tags provided in the treebank. The functional tags are attached to the subordinate clause and they include CND (conditional), PRP (purpose or reason), MNR (manner), or ADV (other types of subordinate clauses that are adjuncts to the main clause).

**Complementation (COMP):** When a comma separates a verb governor and its complement clause, this verb and its subject generally describe the attribution of the complement clause. Attribution is an important notion in discourse analysis in both the RST framework and in the PDT. An example of this is given in (6), and the syntactic pattern used to extract this relation is illustrated in (f).

- (6) 该 公司 介绍 , [8] 在 未来 的  
 The company present , at future DE  
 五 年 内 他们 将 追加 投资  
 five year within they will additionally invest  
 九 千 万 美 元 , [9] 预 计  
 ninety million U.S. dollars , estimate  
 年 产 值 可 达  
 annual output will reach  
 三 亿 美 元 。  
 three hundred million U.S. dollars .

“According to the the company’s presentation, they will invest an additional ninety million

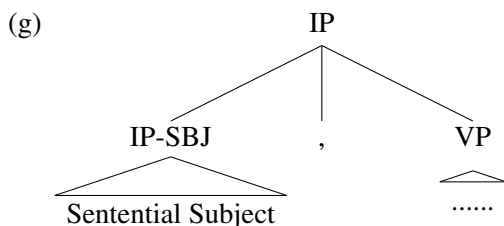
U.S. dollars in the next five years, and the estimated annual output will reach \$ 300 million.”



**Sentential Subject (SBJ):** This category is for commas that separate a sentential subject from its predicate VP. An example is given in (7) and the syntactic pattern used to extract this relation is illustrated in (g).

(7) 出口 快速 增长 , [10] 成为 推动  
 export rapid grow , become promote  
 经济 增长 的 重要 力量 。  
 economy growth DE important force .

“The rapid growth of export becomes an important force in promoting economic growth.”



**Others (OTHER):** The remaining cases of comma receive the OTHER label, indicating they do not mark the boundary of a discourse segment.

Our proposed comma classification scheme serves the dual purpose of identifying elementary discourse units and at the same time detecting coarse-grained discourse relations anchored by the comma. The discourse relations identified in this manner by no means constitute the full discourse analysis of a text, they are, however, a good first approximation. The advantage of our approach is that we do not require manual discourse annotations, and all the information we need is automatically extracted from the syntactic annotation of the CTB and attached to instances of the comma in the corpus. This makes it possible for us to train supervised models to automatically classify the commas in any Chinese text.

### 3 Two comma classification methods

Given the gold standard parses, based on the syntactic patterns described in Section 2, we can map the POS tag of each comma instance in the CTB to one of the seven classes described in Section 2. Using this relabeled data as training data, we experimented with two automatic comma disambiguation methods. In the first method, we simply retrained the Berkeley parser (Petrov and Klein, 2007) on the relabeled data and computed how accurately the commas are labeled in a held-out test set. In the second method, we trained a Maximum Entropy classifier with the Mallet (McCallum et al., 2002) machine learning package to classify the commas. The features are extracted from the CTB data automatically parsed with the Berkeley parser. We implemented features described in (Xue and Yang, 2011), and also experimented with a set of new features as follows. In general, these new features are extracted from the two text spans surrounding the comma. Given a comma, we define the preceding text span as  $i$  span and the following text span as  $j$  span. We also collected a number of subject-predicate pairs from a large corpus that doesn't overlap with the CTB. We refer to this corpus as the auxiliary corpus.

**Subject and Predicate features:** We explored various combinations of the subject ( $sbj$ ), predicate ( $pred$ ) and object ( $obj$ ) of the two spans. The subject of  $i$  span is represented as  $sbj_i$ , etc.

1. The existence of  $sbj_i$ ,  $sbj_j$ , both, or neither.
2. The lemma of  $pred_i$ , the lemma of  $pred_j$ , the conjunction of  $sbj_i$  and  $pred_j$ , the conjunction of  $pred_i$  and  $sbj_j$
3. whether the conjunction of  $sbj_i$  and  $pred_j$  occurs more than 2 times in the auxiliary corpus when  $j$  does not have a subject.
4. whether the conjunction of  $obj_i$  and  $pred_j$  occurs more than 2 times in the auxiliary corpus when  $j$  does not have a subject
5. Whether the conjunction of  $pred_i$  and  $sbj_j$  occurs more than 2 times in the auxiliary corpus when  $i$  does not have a subject.

**Mutual Information features:** Mutual information is intended to capture the association strength between the subject of a previous span and the predicate of the current span. We use Mutual Information

(Church and Hanks, 1989) as shown in Equation (1) and the frequency count computed based on the auxiliary corpus to measure such constraints.

$$MI = \log_2 \frac{\# \text{ co-occur of S and P * corpus size}}{\# \text{ S occur} * \# \text{ P occur}} \quad (1)$$

1. The conjunction of  $sbj_i$  and  $pred_j$  when  $j$  does not have a subject if their  $MI$  value is greater than -8.0, an empirically established threshold.
2. Whether  $obj_i$  and  $pred_j$  has an MI value greater than 5.0 if  $j$  does not have a subject.
3. Whether the MI value of  $sbj_i$  and  $pred_j$  is greater than 0.0, and they occur 2 times in the auxiliary corpus when  $j$  doesn't have a subject.
4. Whether the MI value of  $obj_i$  and  $pred_j$  is greater than 0.0 and they occur 2 times in the auxiliary corpus when  $j$  doesn't have a subject.
5. Whether the MI value of  $pred_i$  and  $sbj_j$  is greater than 0.0 and they occur more than 2 times in the auxiliary corpus when  $i$  does not have a subject.

**Span features:** We used span features to capture syntactic information, e.g. the comma separated spans are constituents in Tree (b) but not in Tree (d).

1. Whether  $i$  forms a single constituent, whether  $j$  forms a single constituent.
2. The conjunction and hierarchical relation of all constituent labels in  $i/j$ , if  $i/j$  does not form a single constituent. The conjunction of all constituent labels in both spans, if neither span form a single constituent.

**Lexical features:**

1. The first word in  $i$  if it is an adverb, the first word in  $j$  if it is an adverb.
2. The first word in  $i$  span if it is a coordinating conjunction, the first word in  $j$  if it is a coordinating conjunction.

## 4 Experiments

### 4.1 Datasets

We use the CTB 6.0 in our experiments and divide it into training, development and test sets using the data split recommended in the CTB 6.0 documentation, as shown in Table 1. There are 5436 commas

in the test set, including 1327 commas that are sentence boundaries (SB), 539 commas that connect coordinated IPs (IP\_COORD), 1173 commas that join coordinated VPs (VP\_COORD), 379 commas that delimit a subordinate clause and its main clause (ADJ), 314 commas that anchor complementation relations (COMP), and 1625 commas that belong to the OTHER category.

### 4.2 Results

As mentioned in Section 3, we experimented with two comma classification methods. In the first method, we replace the part-of-speech (POS) tags of the commas with the seven classes defined in Section 2. We then retrain the Berkeley parser (Petrov and Klein, 2007) using the training set as presented in Table 1, parse the test set, and evaluate the comma classification accuracy.

In the second method, we use the relabeled commas as the gold-standard data to train a supervised classifier to automatically classify the commas. As shown in the previous section, syntactic structures are an important source of information for our classifier. For feature extraction purposes, the entire CTB6.0 is automatically parsed in a round-robin fashion. We divided CTB 6.0 into 10 portions, and parsed each portion with a model trained on other portions, using the Berkeley parser (Petrov and Klein, 2007). Measured by the ParsEval metric (Black et al., 1991), the parsing accuracy on the CTB test set stands at 83.29% (F-score), with a precision of 85.18% and a recall of 81.49%.

The results are presented in Table 2, which shows the overall accuracy of the two methods as well as the results for each individual category. As should be clear from Table 2, the results for the two methods are very comparable, with the second method performing modestly better than the first method.

#### 4.2.1 Subject continuity

One of the goals for this classification scheme is to model subject continuity, which answers the question of how accurately we can predict whether two comma-separated text spans have the same subject or different subjects. When the two spans share the same subject, the comma belongs to the category VP\_COORD. When they have different subjects, they belong to the categories IP\_COORD or

Data	Train	Dev	Test
CTB-6.0	81-325, 400-454, 500-554	41-80	(1-40,901-931 newswire)
	590-596, 600-885, 900	1120-1129	(1018, 1020, 1036, 1044
	1001-1017, 1019, 1021-1035	2140-2159	1060-1061,
	1037-1043, 1045-1059,1062-1071	2280-2294	1072, 1118-1119, 1132
	1073-1078, 1100-1117, 1130-1131	2550-2569	1141-1142, 1148 magazine)
	1133-1140, 1143-1147, 1149-1151	2775-2799	(2165-2180, 2295-2310
	2000-2139, 2160-2164, 2181-2279	3080-3109	2570-2602, 2800-2819
	2311-2549, 2603-2774, 2820-3079		3110-3145 broadcast news)

Table 1: CTB 6.0 data set division.

SB. When this question is meaningless, e.g., when one of the span does not even have a subject, the comma belongs to other categories. To evaluate the performance of our model on this problem, we re-computed the results by putting IP\_COORD and SB in one category, putting VP\_COORD in another category and the rest of the labels in a third category. The results are presented in Table 3.

#### 4.2.2 The effect of genre

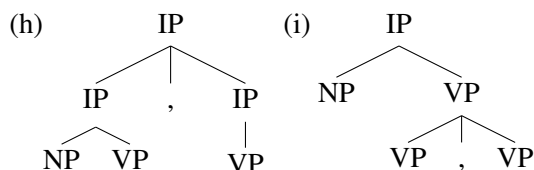
CTB 6.0 consists of data from three different genres, including newswire, magazine and broadcast news. Data genres may have very different characteristics. To evaluate how our model works on different genres, we train a model using training and development sets, and test the model on different genres as described in Table 1. The results on these three genres are presented in Table 4, and they shows a significant fluctuation across genres. Our model works the best on newswire, but not as good on broadcast news and magazine articles.

#### 4.2.3 Comparison with prior work

(Xue and Yang, 2011) presented results on a binary classification of whether or not a comma marks a sentence boundary, while the present work addresses a multi-category classification problem aimed at identifying discourse segments and preliminary discourse relations anchored by the comma. However, since we also have a SB category, comparison is possible. For comparison purposes, we retrained our model on their data sets, and computed the results of SB vs other categories. The results are shown in Table 5. Our results are very comparable with (Xue and Yang, 2011) despite that we are performing a multicategory classification.

#### 4.3 Error analysis

Even though our feature-based approach can theoretically “correct” parsing errors, meaning that a comma can in theory be classified correctly even if a sentence is incorrectly parsed, when examining the system output, errors in automatic parses often lead to errors in comma classification. A common parsing error is the confusion between Structures (h) and (i). If the subject of the text span after a comma is dropped as shown in (h), the parser often produces a VP coordination structure as shown in (i) and vice versa. This kind of parsing errors would lead to errors in our syntactic features and thus directly affect the accuracy of our model.



### 5 Related Work

There is a large body of work on discourse analysis in the field of Natural Language Processing. Most of the work, however, are on English. An unsupervised approach was proposed to recognize discourse relations in (Marcu and Echiabi, 2002), which extracts discourse relations that hold between arbitrary spans of text making use of cue phrases. Like the present work, a lot of research on discourse analysis is carried out at the sentence level. (Soricut and Marcu, 2003; Sporleder and Lapata, 2005; Polanyi et al., 2004). (Soricut and Marcu, 2003) and (Polanyi et al., 2004) implement models to perform discourse parsing, while (Sporleder and Lapata, 2005) introduces discourse chunking as an alternative to full-

Class	Metric	Method 1	Method 2
<i>all</i>	acc. (%)	71.5	72.9
SB	Prec. (%)	65.6	66.2
	Rec. (%)	71.7	73.1
	F. (%)	68.5	69.5
IP_COORD	Prec. (%)	53.3	56.0
	Rec. (%)	50.5	48.6
	F. (%)	52.0	52.0
VP_Coord	Prec. (%)	65.6	68.3
	Rec. (%)	76.3	78.2
	F. (%)	70.5	72.9
ADJ	Prec. (%)	66.9	66.8
	Rec. (%)	29.3	37.7
	F. (%)	40.8	48.2
Comp	Prec. (%)	88.3	91.2
	Rec. (%)	93.9	92.4
	F. (%)	91.0	91.8
SentSBJ	Prec. (%)	25.0	31.8
	Rec. (%)	6	10
	F. (%)	9.7	15.6
Other	Prec. (%)	86.9	85.6
	Rec. (%)	83.4	84.1
	F. (%)	85.1	84.8

Table 2: Overall accuracy of the two methods as well as the results for each individual category.

scale discourse parsing.

The emergence of linguistic corpora annotated with discourse structure such as the RST Discourse Treebank (Carlson et al., 2002) and PDT (Miltsakaki et al., 2004; Prasad et al., 2008) have changed the landscape of discourse analysis. More robust, data-driven models are starting to emerge.

Compared with English, much less work has been done in Chinese discourse analysis, presumably due to the lack of discourse resources in Chinese. (Huang and Chen, 2011) constructs a small corpus following the PDT annotation scheme and

	Prec. (%)	Rec. (%)	F. (%)
VP_COORD	68.3	78.2	72.9
IP_COORD+SB	76.0	78.7	77.3
Other	89.0	80.2	84.4

Table 3: Subject continuity results based on Maximum Entropy model

Genre	NW	BN	MZ
Accuracy. (%)	79.1	73.6	67.7

Table 4: Results on different genres based on Maximum Entropy model

	Xue and Yang			our model		
(%)	p	r	f1	p	r	f1
Overall			89.2			88.7
EOS	64.7	76.4	70.1	63.0	77.9	69.7
NEOS	95.1	91.7	93.4	95.3	90.8	93.0

Table 5: Comparison of (Xue and Yang, 2011) and the present work based on Maximum Entropy model

trains a statistical classifier to recognize discourse relations. Their work, however, is only concerned with discourse relations between adjacent sentences, thus side-stepping the hard problem of disambiguating the Chinese comma and analyzing intra-sentence discourse relations. To the best of our knowledge, our work is the first in attempting to disambiguating the Chinese comma as the first step in performing Chinese discourse analysis.

## 6 Conclusions and future work

We proposed a approach to disambiguate the Chinese comma as a first step toward discourse analysis. Training and testing data are automatically derived from a syntactically annotated corpus. We presented two automatic comma disambiguation methods that perform comparably. In the first method, comma disambiguation is integrated into the parsing process while in the second method we train a supervised classifier to classify the Chinese comma, using features extracted from automatic parses. Much needs to be done in the area, but we believe our work provides insight into the intricacy and complexity of discourse analysis in Chinese.

## Acknowledgment

This work is supported by the IIS Division of National Science Foundation via Grant No. 0910532 entitled “Richer Representations for Machine Translation”. All views expressed in this paper are those of the authors and do not necessarily represent the view of the National Science Foundation.

## References

- L Carlson, D Marcu, M E Okurowski. 2002. *RST Discourse Treebank*. Linguistic Data Consortium 2002.
- Caroline Sporleder, Mirella Lapata. 2005. *Discourse chunking and its application to sentence compression*. In Proceedings of HLT/EMNLP 2005.
- Livia Polanyi, Chris Culy, Martin Van Den Berg, Gian Lorenzo Thione and David Ahn. 2004. *Sentential structure and discourse parsing*. In Proceedings of the ACL 2004 Workshop on Discourse Annotation 2004.
- Hen-Hsen Huang and Hsin-Hsi Chen. 2011. *Chinese Discourse Relation Recognition*. In Proceedings of the 5th International Joint Conference on Natural Language Processing 2011, pages 1442-1446.
- Daniel Marcu and Abdessamad Echihabi. 2002. *An Unsupervised Approach to Recognizing Discourse Relations*. In Proceedings of the ACL, July 6-12, 2002, Philadelphia, PA, USA.
- Radu Soricut and Daniel Marcu. 2003. *Sentence Level Discourse Parsing using Syntactic and Lexical Information*. In Proceedings of the ACL 2003.
- Eleni Miltsakaki, Rashmi Prasad, Aravind Joshi and Bonnie Webber. 2004. *The Penn Discourse Treebank*. In Proceedings of LREC 2004.
- Nianwen Xue and Yaqin Yang. 2011. *Chinese sentence segmentation as comma classification*. In Proceedings of ACL 2011.
- Nianwen Xue, Fei Xia, Fu-Dong Chiou and Martha Palmer. 2005. *The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus*. Natural Language Engineering, 11(2):207-238.
- Slav Petrov and Dan Klein. 2007. *Improved Inferencing for Unlexicalized Parsing*. In Proceedings of HLT-NAACL 2007.
- E. Black, S. Abney, D. Flickinger, C. Gdaniec, R. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. *A procedure for quantitatively comparing the syntactic coverage of English grammars*. In Proceedings of the DARPA Speech and Natural Language Workshop, pages 306-311.
- Mann, William C. and Sandra A. Thompson. 1988. *Rhetorical Structure Theory: Toward a functional theory of text organization*. Text 8 (3): 243-281.
- Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. 2008. *The Penn Discourse Treebank 2.0.*. In Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008).
- Meixun Jin, Mi-Young Kim, Dong-Il Kim, and Jong-Hyeok Lee. 2004. *Segmentation of Chinese Long Sentences Using Commas*. In Proceedings of the SIGHANN Workshop on Chinese Language Processing.
- Xing Li, Chengqing Zong, and Rile Hu. 2005. *A Hierarchical Parsing Approach with Punctuation Processing for Long Sentence Sentences*. In Proceedings of the Second International Joint Conference on Natural Language Processing: Companion Volume including Posters/Demos and Tutorial Abstracts.
- Andrew Kachites McCallum. 2002. *MALLET: A Machine Learning for Language Toolkit*. <http://mallet.cs.umass.edu>.
- Church, K., and Hanks, P. 1989. *Word Association Norms, Mutual Information and Lexicography*. Association for Computational Linguistics, Vancouver , Canada

# Collective Classification for Fine-grained Information Status

Katja Markert<sup>1,2</sup>, Yufang Hou<sup>2</sup>, Michael Strube<sup>2</sup>

<sup>1</sup> School of Computing, University of Leeds, UK, `sckm@leeds.ac.uk`

<sup>2</sup> Heidelberg Institute for Theoretical Studies gGmbH, Heidelberg, Germany  
(`yufang.hou|michael.strube`)@h-its.org

## Abstract

Previous work on classifying information status (Nissim, 2006; Rahman and Ng, 2011) is restricted to coarse-grained classification and focuses on conversational dialogue. We here introduce the task of classifying fine-grained information status and work on written text. We add a fine-grained information status layer to the Wall Street Journal portion of the OntoNotes corpus. We claim that the information status of a mention depends not only on the mention itself but also on other mentions in the vicinity and solve the task by collectively classifying the information status of all mentions. Our approach strongly outperforms reimplementations of previous work.

## 1 Introduction

Speakers present already known and yet to be established information according to principles referred to as *information structure* (Prince, 1981; Lambrecht, 1994; Kruijff-Korbayová and Steedman, 2003, inter alia). While information structure affects all kinds of constituents in a sentence, we here adopt the more restricted notion of *information status* which concerns only discourse entities realized as noun phrases, i.e. *mentions*<sup>1</sup>. Information status (*IS* henceforth) describes the degree to which a discourse entity is available to the hearer with regard to the speaker’s assumptions about the hearer’s knowledge and beliefs (Nissim et al., 2004). Old mentions are known to the hearer and have been referred

<sup>1</sup>Since not all noun phrases are referential, we call noun phrases which carry information status *mentions*.

to previously. Mediated mentions have not been mentioned before but are also not autonomous, i.e., they can only be correctly interpreted by reference to another mention or to prior world knowledge. All other mentions are *new*.

IS can be beneficial for a number of NLP tasks, though the results have been mixed. Nenkova et al. (2007) used IS as a feature for generating pitch accent in conversational speech. As IS is restricted to noun phrases, while pitch accent can be assigned to any word in an utterance, the experiments were not conclusive. For determining constituent order of German sentences, Cahill and Riester (2009) incorporate features modeling IS to good effect. Rahman and Ng (2011) showed that IS is a useful feature for coreference resolution.

Previous work on learning IS (Nissim, 2006; Rahman and Ng, 2011) is restricted in several ways. It deals with conversational dialogue, in particular with the corpus annotated by Nissim et al. (2004). However, many applications that can profit from IS concentrate on written texts, such as summarization. For example, Siddharthan et al. (2011) show that solving the IS subproblem of whether a person proper name is already known to the reader improves automatic summarization of news. Therefore, we here model IS in written text, creating a new dataset which adds an IS layer to the already existing comprehensive annotation in the OntoNotes corpus (Weischedel et al., 2011). We also report the first results on fine-grained IS classification by modelling further distinctions within the category of mediated mentions, such as comparative and bridging anaphora (see Examples 1 and 2, re-

spectively).<sup>2</sup> Fine-grained IS is a prerequisite to full bridging/comparative anaphora resolution, and therefore necessary to fill gaps in entity grids (Barzilay and Lapata, 2008) based on coreference only. Thus, Examples 1 and 2 do not exhibit any coreferential entity coherence but coherence can be established when the comparative anaphor *others* is resolved to *others than freeway survivor Buck Helm*, and the bridging anaphor *the streets* is resolved to *the streets of Oranjemund*, respectively.

- (1) the condition of *freeway survivor Buck Helm* . . . , improved, hospital officials said. Rescue crews, however, gave up hope that **others** would be found.
- (2) *Oranjemund, the mine headquarters*, is a lonely corporate oasis of 9,000 residents. Jackals roam **the streets** at night . . .

We approach the challenge of modeling IS via collective classification, using several novel linguistically motivated features. We reimplement Nissim’s (2006) and Rahman and Ng’s (2011) approaches as baselines and show that our approach outperforms these by a large margin for both coarse- and fine-grained IS classification.

## 2 Related Work

**IS annotation schemes and corpora.** We enhance the approach in Nissim et al. (2004) in two major ways (see also Section 3.1). First, comparative anaphora are not specifically handled in Nissim et al. (2004) (and follow-on work such as Ritz et al. (2008) and Riester et al. (2010)), although some of them might be included in their respective *bridging* subcategories. Second, we apply the annotation scheme reliably to a new genre, namely news. This is a non-trivial extension: Ritz et al. (2008) applied a variation of the Nissim et al. (2004) scheme to a small set of 220 NPs in a German news/commentary corpus but found that reliability then dropped significantly to the range of  $\kappa = 0.55$  to 0.60. They attributed this to the higher syntactic complexity and semantic vagueness in the commentary corpus. Riester et al. (2010) annotated a

<sup>2</sup>All examples in this paper are from the OntoNotes corpus. The mention in question is typed in boldface; antecedents, where applicable, are displayed in italics.

German news corpus marginally reliable ( $\kappa = 0.66$ ) for their overall scheme but their confusion matrix shows even lower reliability for several subcategories, most importantly deixis and bridging.

While standard coreference corpora do not contain IS annotation, some corpora annotated for bridging are emerging (Poesio, 2004; Korzen and Buch-Kromann, 2011) but they are (i) not annotated for comparative anaphora or other IS categories, (ii) often not tested for reliability or reach only low reliability, (iii) often very small (Poesio, 2004).

To the best of our knowledge, we therefore present the first English corpus reliably annotated for a wide range of IS categories as well as full anaphoric information for three main anaphora types (coreference, bridging, comparative).

**Automatic recognition of IS.** Vieira and Poesio (2000) describe heuristics for processing definite descriptions in news text. As their approach is restricted to definites, they only analyse a subset of the mentions we consider carrying IS. Siddharthan et al. (2011) also concentrate on a subproblem of IS only, namely the hearer-old/hearer-new distinctions for person proper names.

Nissim (2006) and Rahman and Ng (2011) both present algorithms for IS detection on Nissim et al.’s (2004) Switchboard corpus. Both papers treat IS classification as a local classification problem whereas we look at dependencies between the IS status of different mentions, leading to collective classification. In addition, they only distinguish the three main categories *old*, *mediated* and *new*. Finally, we work on news corpora which poses different problems from dialogue.

Anaphoricity determination (Ng, 2009; Zhou and Kong, 2009) identifies many or most *old* mentions. However, no distinction between *mediated* and *new* mentions is made. Most approaches to bridging resolution (Meyer and Dale, 2002; Poesio et al., 2004) or comparative anaphora (Modjeska et al., 2003; Markert and Nissim, 2005) address only the selection of the antecedent for the bridging/comparative anaphor, not its recognition. Sasano and Kurohashi (2009) do also tackle bridging recognition, but they depend on language-specific non-transferrable features for Japanese.



### 3 Corpus Creation

#### 3.1 Annotation Scheme

Our scheme follows Nissim et al. (2004) in distinguishing three major IS categories *old*, *new* and *mediated*. A mention is *old* if it is either coreferential with an already introduced entity or a generic or deictic pronoun. We follow the OntoNotes (Weischedel et al., 2011) definition of coreference to be able to integrate our annotations with it. This definition includes coreference with noun phrase as well as verb phrase antecedents<sup>3</sup>.

*Mediated* refers to entities which have not yet been introduced in the text but are inferrable via other mentions or are known via world knowledge. We distinguish the following six subcategories: The category *mediated/comparative* comprises mentions compared via either a contrast or similarity to another one (see Example 1). This category is novel in our scheme. We also include a category *mediated/bridging* (see Examples 2, 3 and 4). Bridging anaphora can be any noun phrase and are not limited to definite NPs as in Poesio et al. (2004), Gardent and Manuélian (2005), Riester et al. (2010). In contrast to Nissim et al. (2004), antecedents for both comparative and bridging categories are annotated and can be noun phrases, verb phrases or even clauses. The category *mediated/knowledge* is inspired by the hearer-old distinction introduced by Prince (1992) and covers entities generally known to the hearer. It includes many proper names, such as *Poland*.<sup>4</sup> Mentions that are syntactically linked via a possessive relation or a PP modification to other, *old* or *mediated* mentions fall into the type *mediated/synt* (see Examples 5 and 6).<sup>5</sup> With no change to Nissim et al.'s scheme, coordinated mentions where at least one element in the conjunction is *old* or *mediated* are covered by the category *mediated/aggregate*, and mentions referring to a value of a previously mentioned function by the type *mediated/func*.

All other mentions are annotated as *new*, includ-

<sup>3</sup>In contrast to Nissim et al. (2004), but in accordance with OntoNotes, we do not consider generics for coreference.

<sup>4</sup>This class corresponds roughly to Nissim et al.'s (2004) *mediated/general*.

<sup>5</sup>This class expands Nissim et al.'s (2004) *poss* category that only considers possessives but not PP modification.

ing most generics as well as newly introduced, specific mentions such as Example 7.

- (3) Initial steps were taken at *Poland's first environmental conference, which I attended last month*. ... it was no accident that **participants** urged the free flow of information
- (4) The Bakersfield supermarket *went out of business* last May. **The reason** was ...
- (5) One Washington couple sold *their liquor store*
- (6) *the main artery into San Francisco*
- (7) the owner was murdered by *robbers*

#### 3.2 Agreement Study

We carried out an agreement study with 3 annotators, of which Annotator A was the scheme developer and first author of this paper. All texts used were from the Wall Street Journal (WSJ) portion of OntoNotes. There were no restrictions on which texts to include apart from (i) exclusion of letters to the editor as they contain cross-document links and (ii) a preference for longer texts with potentially richer discourse structure.

Mentions were automatically preselected for the annotators using the gold-standard syntactic annotation.<sup>6</sup> The existing coreference annotation was automatically carried over to the IS task by marking all mentions in a coreference chain (apart from the first mention in the chain) as *old*. The annotation task consisted of marking all mentions for their IS (*old*, *mediated* or *new*) as well as marking *mediated* subcategories (see Section 3.1) and the antecedents for comparative and bridging anaphora.

The scheme was developed on 9 texts, which were also used for training the annotators. Inter-annotator agreement was measured on 26 new texts, which included 5905 pre-marked potential mentions. The annotations of 1499 of these were carried over from OntoNotes, leaving 4406 potential mentions for annotation and agreement measurement. In addition to

<sup>6</sup>Some non-mentions such as idioms could not be filtered out via the syntactic annotation and had to be excluded during human annotation.

	A-B	A-C	B-C
Overall Percentage coarse	87.5	86.3	86.5
Overall $\kappa$ coarse	77.3	75.2	74.7
Overall Percentage fine	86.6	85.3	85.7
Overall $\kappa$ fine	80.1	77.7	77.3

Table 1: Agreement Results

	A-B	A-C	B-C
$\kappa$ Non-mention	81.5	78.9	86.0
$\kappa$ Old	80.5	83.2	79.3
$\kappa$ New	76.6	74.0	74.3
$\kappa$ Mediated/Knowledge	82.1	78.4	74.1
$\kappa$ Mediated/Synt	88.4	87.8	87.6
$\kappa$ Mediated/Aggregate	87.0	85.4	86.0
$\kappa$ Mediated/Func	6.0	83.2	6.9
$\kappa$ Mediated/Comp	81.8	78.3	81.2
$\kappa$ Mediated/Bridging	70.8	60.6	62.3

Table 2: Agreement Results for individual categories

percentage agreement, we measured Cohen’s  $\kappa$  (Artstein and Poesio, 2008) between all 3 possible annotator pairings. We also report single-category agreement for each category, where all categories but one are merged and then  $\kappa$  is computed as usual. Table 1 shows agreement results for the overall scheme at the coarse-grained (4 categories: *non-mention*, *old*, *new*, *mediated*) and the fine-grained level (9 categories: *non-mention*, *old*, *new* and the 6 *mediated* subtypes). The results show that the scheme is overall reliable, with not too many differences between the different annotator pairings.<sup>7</sup>

Table 2 shows the individual category agreement for all 9 categories. We achieve high reliability for most categories.<sup>8</sup> Particularly interesting is the fact that hearer-old entities (*mediated/knowledge*) can be identified reliably although all annotators had substantially different backgrounds. The reliability of the category *bridging* is more annotator-dependent, although still higher, sometimes considerably, than other previous attempts at bridg-

<sup>7</sup>Often, annotation is considered highly reliable when  $\kappa$  exceeds 0.80 and marginally reliable when between 0.67 and 0.80 (Carletta, 1996). However, the interpretation of  $\kappa$  is still under discussion (Artstein and Poesio, 2008).

<sup>8</sup>The low reliability of the rare category *func*, when involving Annotator B, was explained by Annotator B forgetting about this category after having used it once. Pair A-C achieved high reliability ( $\kappa$  83.2 for pair A-C).

ing annotation (Poesio et al., 2004; Gardent and Manuélian, 2005; Riestler et al., 2010).

### 3.3 Gold Standard

Our final gold standard corpus consists of 50 texts from the WSJ portion of the OntoNotes corpus. The corpus will be made publically available as OntoNotes annotation layer via <http://www.h-its.org/nlp/download>.

Disagreements in the 35 texts used for annotator training (9 texts) and testing (26 texts) were resolved via discussion between the annotators. An additional 15 texts were annotated by Annotator A. Finally, Annotator A carried out consistency checks over all texts. – The gold standard includes 10,980 true mentions (see Table 3).

Texts	50
Mentions	10,980
old	3237
coref	3,143
generic_deictic_pr	94
mediated	3,708
world knowledge	924
syntactic	1,592
aggregate	211
func	65
comparative	253
bridging	663
new	4,035

Table 3: Gold Standard Distribution

## 4 Features

In this Section, we describe both the local as well as the relational features we use.

### 4.1 Features for Local Classification

We use the following local features, including the features in Nissim (2006) and Rahman and Ng (2011) to be able to gauge how their systems fare on our corpus and as a comparison point for our novel collective classification approach.

The features developed by Nissim (2006) are shown in Table 4. Nissim shows clearly that these features are useful for IS classification. Thus, subjects are more likely to be *old* as assumed by, e.g., centering theory (Grosz et al.,

Feature	Value
full prev mention	{yes, no, NA} <sup>9</sup>
mention time	{first, second, more}
partial prev mention	{yes, no, NA}
determiner	{bare, def, dem, indef, poss, NA}
NP type	{pronoun, common, proper, other}
NP length	numeric
grammatical role	{subject, subjpass, pp, other}

Table 4: Nissim’s (2006) feature set

1995). Also, previously unmentioned proper names are more likely to be hearer-old and therefore mediated/knowledge, although their exact status will depend on how well known a particular proper name is.

Rahman and Ng (2011) add all *unigrams* appearing in any mention in the training set as features. They also integrated (via a convolution tree-kernel SVM (Collins and Duffy, 2001)) partial parse trees that capture the generalised syntactic context of a mention  $e$  and include the mention’s parent and sibling nodes without lexical leaves. However, they use no structure underneath the mention node  $e$  itself, assuming that “any NP-internal information has presumably been captured by the flat features”.

To these feature sets, we add a small set of other local features *otherlocal*. These track partial previous mentions by also counting partial previous mention time as well as the previous mention of content words only. We also add a mention’s number as one of singular, plural or unknown, and whether the mention is modified by an adjective. Another feature encapsulates whether the mention is modified by a comparative marker, using a small set of 10 markers such as *another*, *such*, *similar* ... and the presence of adjectives or adverbs in the comparative. Finally, we include the mention’s semantic class as one of 12 coarse-grained classes, including location, organisation, person and several classes for numbers (such as date, money or percent).

## 4.2 Relations for Collective Classification

Both Nissim (2006) and Rahman and Ng (2011) classify each mention individually in a standard supervised ML setting, not considering potential dependencies between the IS categories of different

<sup>9</sup>We changed the value of “full prev mention” from “numeric” to {yes, no, NA}.

mentions. However, collective or joint classification has made substantial impact in other NLP tasks, such as opinion mining (Pang and Lee, 2004; Somasundaran et al., 2009), text categorization (Yang et al., 2002; Taskar et al., 2002) and the related task of coreference resolution (Denis and Baldridge, 2007). We investigate two types of relations between mentions that might impact on IS classification.

**Syntactic parent-child relations.** Two mediated subcategories account for accessibility via syntactic links to another old or mediated mention: *mediated/synt* is used when at least one child of a mention is *mediated* or *old*, with child relations restricted to pre- or postnominal possessives as well as PP children in our scheme (see Section 3.1). *mediated/aggregate* is for coordinations in which at least one of the children is *old* or *mediated*. In these two cases, a mention’s IS depends directly on the IS of its children. We therefore link a mention  $m_1$  to a mention  $m_2$  via a *hasChild* relation if (i)  $m_2$  is a possessive or prepositional modification of  $m_1$ , or (ii)  $m_1$  is a coordination and  $m_2$  is one of its children.

Using such a relational feature catches two birds with one stone: firstly, it integrates the internal structure of a mention into the algorithm, which Rahman and Ng (2011) ignore; secondly, it captures dependencies between parent and child classification, which would not be possible if we integrated the internal structure via flat features or additional tree kernels. We hypothesise that the higher syntactic complexity of our news genre (14.5% of all mentions are *mediated/synt*) will make this feature highly effective in distinguishing between *new* and *mediated* categories.

**Syntactic precedence relations.** IS is said to influence word order (Birner and Ward, 1998; Cahill and Riester, 2009) and this fact has been exploited in work on generation (Prevost, 1996; Filippova and Strube, 2007; Cahill and Riester, 2009). Therefore, we integrate dependencies between the IS classification of mentions in precedence relations.

$m_1$  precedes  $m_2$  if (i)  $m_1$  and  $m_2$  are in the same clause, allowing for trace subjects in gerund and infinitive constructions, (ii)  $m_1$  and  $m_2$  are dependent on the same verb or noun, allowing for intervening nodes via modal, auxiliary, gerund and infinitive

constructions, (iii)  $m_1$  is neither a child nor a parent of  $m_2$ , and (iv)  $m_1$  occurs before  $m_2$ .

For Example 8 (slightly simplified) we extract the precedence relations shown in Table 5.

- (8) She was sent by her mother to a white woman’s house to do chores in exchange for meals and a place to sleep.

(She)<sub>old</sub> ><sub>p</sub> (her mother)<sub>med/synt</sub>  
 (She)<sub>old</sub> ><sub>p</sub> (a white-woman’s house)<sub>new</sub>  
 (She)<sub>old</sub> ><sub>p</sub> (chores)<sub>new</sub>  
 (She)<sub>old</sub> ><sub>p</sub> (exchange .....sleep)<sub>new</sub>  
 (her mother)<sub>med/synt</sub> ><sub>p</sub> (a white woman’s house)<sub>new</sub>  
 (chores)<sub>new</sub> ><sub>p</sub> (exchange ... sleep)<sub>new</sub>  
 (meals)<sub>new</sub> ><sub>p</sub> (a place to sleep)<sub>new</sub>

Table 5: Precedence Relations for Example 8. *She* is a trace subject for *do*.

Proper names behave differently from common nouns. For example, they can occur at many different places in the clause when functioning as spatial or temporal scene-setting elements, such as *In New York*. We therefore exclude all precedence relations where one element of the pair is a proper name.

We extract 2855 precedence relations. Table 6 shows the statistics on precedence with the first mention in a pair in rows and the second in columns. Mediated and new mentions indeed rarely precede old mentions, so that precedence should improve separating of `old` vs other mentions.

	old	mediated	new
old	136	387	519
mediated	88	357	379
new	85	291	613

Table 6: Precedence relations in our corpus

## 5 Experiments

### 5.1 Experimental Setup

We use our gold standard corpus (see Section 3.3) via 10-fold cross-validation on documents for all experiments. Following Nissim (2006) and Rahman and Ng (2011), we perform all experiments on gold standard mentions and use the human WSJ syntactic annotation for feature extraction, when necessary. For the extraction of semantic class, we use

OntoNotes entity type annotation for proper names and an automatic assignment of semantic class via WordNet hypernyms for common nouns.

Coarse-grained versions of all algorithms distinguish only between the three `old`, `mediated`, `new` categories. Fine-grained versions distinguish between the categories `old`, the six `mediated` subtypes, and `new`. We report overall accuracy as well as precision, recall and F-measure per category. Significance tests are conducted using McNemar’s test on overall algorithm accuracy, at the level of 1%.

### 5.2 Local Classifiers

We reimplemented the algorithms in Nissim (2006) and Rahman and Ng (2011) as comparison baselines, using their feature and algorithm choices. Algorithm *Nissim* is therefore a decision tree J48 with standard settings in WEKA with the features in Table 4. Algorithm *RahmanNg* is an SVM with a composite kernel and one-vs-all training/testing (toolkit SVMLight). They use the features in Table 4 plus unigram and tree kernel features, described in Section 4.1. We add our additional set of *otherlocal* features to both baseline algorithms (yielding *Nissim+ol* and *RahmanNg+ol*) as they aim specifically at improving fine-grained classification.

### 5.3 Collective Classification

For incorporating our inter-mention links, we use a variant of Iterative Collective classification (ICA), which has shown good performance over a variety of tasks (Lu and Getoor, 2003) and has been used in NLP for example for opinion mining (Somasundaran et al., 2009). ICA is normally faster than Gibbs sampling and — in initial experiments — did not yield significantly different results from it.

ICA initializes each mention with its most likely IS, according to the local classifier and features. It then iterates a relational classifier, which uses both local and relational features (our *hasChild* and *precedes* features) taking IS assignments to neighbouring mentions into account. We use the *exist* aggregator to define the dependence between mentions.

We use NetKit (Macskassy and Provost, 2007) with its standard ICA settings for collective inference, as it allows direct comparison between local and collective classification. The relational classifiers are always exactly the same classifiers as the

	<i>local</i>						<i>collective</i>					
	<i>Nissim</i>			<i>Nissim+ol</i>			<i>Nissim+ol+hasChild</i>			<i>Nissim+ol+hasChild+precedes</i>		
	R	P	F	R	P	F	R	P	F	R	P	F
Coarse												
old	82.2	86.4	84.2	81.2	88.6	84.8	81.7	88.6	<b>85.0</b>	80.9	89.1	84.8
mediated	51.9	60.2	55.7	57.8	64.6	61.0	68.4	77.4	<b>72.6</b>	68.8	76.9	<b>72.6</b>
new	74.2	63.6	68.5	78.4	67.3	72.4	87.7	75.1	<b>80.9</b>	87.9	75.0	<b>80.9</b>
acc		69.0			72.3			<b>79.4</b>			<b>79.4</b>	
Fine												
old	84.0	83.3	83.6	85.0	83.9	84.5	84.3	84.7	84.5	84.1	85.2	<b>84.6</b>
med/knowledge	61.3	60.0	60.6	61.0	69.5	65.0	62.3	70.0	<b>65.9</b>	60.6	70.0	65.0
med/synt	37.2	59.7	45.8	44.7	60.0	51.3	76.8	81.4	<b>79.0</b>	75.7	80.1	77.9
med/agg	26.0	42.0	32.2	20.4	38.4	26.6	42.6	55.9	48.4	43.1	55.8	<b>48.7</b>
med/func	0.0	NA	NA	32.3	65.6	43.3	33.8	53.7	41.5	35.4	53.5	<b>48.7</b>
med/comp	0.4	7.70	0.7	79.0	82.6	80.0	80.6	82.9	<b>81.8</b>	81.4	82.0	81.7
med/bridging	6.6	26.2	10.6	8.9	30.9	13.8	9.6	34.4	15.1	12.2	41.7	<b>18.9</b>
new	82.6	61.0	70.2	82.7	65.1	72.8	88.0	74.0	<b>80.4</b>	87.7	73.3	79.8
acc		66.6			70.0			<b>77.0</b>			76.8	

Table 7: Collective classification compared to Nissim’s local classifier. Best performing algorithms are bolded.

local ones with the relational features added: thus, if the local classifier is a tree kernel SVM so is the relational one. One problem when using the SVM Tree kernel as relational classifier is that it allows only for binary classification so that we need to train several binary networks in a one-vs-all paradigm (see also (Rahman and Ng, 2011)), which will not be able to use the multiclass dependencies of the relational features to optimum effect.

#### 5.4 Results

Table 7 shows the comparison of collective classification to local classification, using Nissim’s framework and features, and Table 8 the equivalent table for Rahman and Ng’s approach.

The improvements using the additional local features over the original local classifiers are statistically significant in all cases. In particular, the inclusion of semantic classes improves mediated/knowledge and mediated/func, and comparative anaphora are recognised highly reliably via a small set of comparative markers.

The *hasChild* relation leads to significant improvement in accuracy over local classification in all cases, showing the value of collective classification. The improvement here is centered on the categories of mediated/synt (for both cases) and mediated/aggregate (for *Nissim+ol+hasChild*) as well as their distinction from

new.<sup>10</sup> It is also interesting that collective classification with a concise feature set and a simple decision tree as used in *Nissim+ol+hasChild*, performs equally well as *RahmanNg+ol+hasChild*, which uses thousands of unigram and tree features and a more sophisticated local classifier. It also shows more consistent improvements over all fine-grained classes.

The *precedes* relation does not lead to any further improvement. We investigated several variations of the precedence link, such as restricting it to certain grammatical relations, taking into account definiteness or NP type but none of them led to any improvement. We think there are two reasons for this lack of success. First, the precedence of mediated vs. new mentions does not follow a clear order and is therefore not a very predictive feature (see Table 6). At first, this seems to contradict studies such as Cahill and Riester (2009) that find a variety of precedences according to information status. However, many of the clearest precedences they find are more specific variants of the  $\text{old} >_p \text{mediated}$  or  $\text{old} >_p \text{new}$  precedence or they are preferences at an even finer level than the one we annotate, including for example the identification of generics. Second, the clear  $\text{old} >_p \text{mediated}$

<sup>10</sup>For *RahmanNg+ol+hasChild*, the aggregate class suffers from collective classification. We hypothesise that this is an artefact of the one-vs-all training/testing for rare categories.

	<i>local</i>						<i>collective</i>					
	<i>RahmanNg</i>			<i>RahmanNg+ol</i>			<i>RahmanNg+ol +hasChild</i>			<i>RahmanNg+ol +hasChild+precedes</i>		
	R	P	F	R	P	F	R	P	F	R	P	F
<b>Coarse</b>												
old	81.3	90.1	85.5	82.6	91.4	<b>86.8</b>	83.5	87.8	85.6	82.9	87.2	85.0
mediated	61.4	68.6	64.8	61.5	71.9	66.3	66.7	79.5	<b>72.6</b>	64.8	76.7	70.3
new	82.1	69.9	75.5	84.9	70.1	76.8	89.0	74.9	<b>81.3</b>	86.9	73.5	79.6
acc	74.9			76.3			<b>79.8</b>			78.3		
<b>Fine</b>												
old	85.1	87.0	86.0	85.6	87.9	<b>86.7</b>	85.3	87.4	86.3	85.8	87.5	86.4
med/knowledge	65.8	67.2	66.5	64.8	72.6	68.5	67.1	69.6	68.3	64.7	73.2	<b>68.7</b>
med/synt	55.8	72.1	62.9	55.8	72.6	63.1	79.8	78.1	<b>78.9</b>	79.8	78.1	<b>78.9</b>
med/agg	29.9	75.9	<b>42.9</b>	29.9	75.9	<b>42.9</b>	17.1	53.7	25.9	14.2	49.2	22.1
med/func	27.7	38.3	32.1	38.5	69.4	<b>49.5</b>	40.0	44.1	42.0	40.0	40.0	40.0
med/comp	25.3	86.5	39.1	76.7	82.2	<b>79.3</b>	74.3	62.7	68.0	74.3	62.7	68.0
med/bridging	10.6	44.6	<b>17.1</b>	9.0	47.2	15.2	1.0	15.2	2.0	1.0	13.7	1.9
new	87.3	66.3	75.4	89.0	67.8	77.0	89.2	74.6	<b>81.2</b>	89.2	74.6	<b>81.2</b>
acc	72.6			74.6			<b>77.5</b>			77.4		

Table 8: Collective classification compared to Rahman and Ng’s local classifier. Best performing algorithms are bolded.

and  $\text{old} >_p \text{new}$  preferences are partially already captured by the local features, especially the grammatical role, as, for example, subjects are often both old as well as early on in a sentence.

With regard to fine-grained classification, many categories including comparative anaphora, are identified quite reliably, especially in the multiclass classification setting (*Nissim+ol+hasChild*). Bridging seems to be the by far most difficult category to identify with final best F-measures still very low. Most bridging mentions do not have any clear internal structure or external syntactic contexts that signal their presence. Instead, they rely more on lexical and world knowledge for recognition. Unigrams could potentially encapsulate some of this lexical knowledge but — without generalization — are too sparse for a relatively rare category such as bridging (6% of all mentions) to perform well. The difficulty of bridging recognition is an important insight of this paper as it casts doubt on the strategy in previous research to concentrate almost exclusively on antecedent selection (see Section 2).

## 6 Conclusions

We presented a new approach to information status classification in written text, for which we also provide the first reliably annotated English language corpus. Based on linguistic intuition, we define fea-

tures for classifying mentions collectively. We show that our collective classification approach outperforms the state-of-the-art in coarse-grained IS classification by about 10% (Nissim, 2006) and 5% (Rahman and Ng, 2011) accuracy. The gain is almost entirely due to improvements in distinguishing between *new* and *mediated* mentions. For the latter, we also report the – to our knowledge – first fine-grained IS classification results.

Since the work reported in this paper relied – following Nissim (2006) and Rahman and Ng (2011) – on gold standard mentions and syntactic annotations, we plan to perform experiments with predicted mentions as well. We also have to improve the recognition of bridging, ideally combining recognition and antecedent selection for a complete resolution component. In addition, we plan to integrate IS resolution with our coreference resolution system (Cai et al., 2011) to provide us with a more comprehensive discourse processing system.

**Acknowledgements.** Katja Markert received a Fellowship for Experienced Researchers by the Alexander-von-Humboldt Foundation and Yufang Hou is funded by a PhD scholarship from the Research Training Group *Coherence in Language Processing* at Heidelberg University. We thank the Heidelberg Institute for Theoretical Studies for hosting Katja Markert and funding the annotation study, and the annotators for their diligent work.

## References

- Ron Artstein and Massimo Poesio. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.
- Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34.
- Betty J. Birner and Gregory Ward. 1998. *Information Status and Noncanonical Word Order in English*. John Benjamins, Amsterdam, The Netherlands.
- Aoife Cahill and Arndt Riester. 2009. Incorporating information status into generation ranking. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing*, Singapore, 2–7 August 2009, pages 817–825.
- Jie Cai, Éva Mújdricza-Maydt, and Michael Strube. 2011. Unrestricted coreference resolution via global hypergraph partitioning. In *Proceedings of the Shared Task of the 15th Conference on Computational Natural Language Learning*, Portland, Oreg., 23–24 June 2011, pages 56–60.
- Jean Carletta. 1996. Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.
- Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems 14*, Vancouver, B.C., Canada, 3–8 December, 2001, pages 625–632, Cambridge, Mass. MIT Press.
- Pascal Denis and Jason Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, N.Y., 22–27 April 2007, pages 236–243.
- Katja Filippova and Michael Strube. 2007. Generating constituent order in German clauses. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic, 23–30 June 2007, pages 320–327.
- Claire Gardent and H el ene Manu elien. 2005. Cr eation d’un corpus annot e pour le traitement des descriptions d efinies. *Traitement Automatique des Langues*, 46(1):115–140.
- Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.
- Iorn Korzen and Matthias Buch-Kromann. 2011. Anaphoric relations in the Copenhagen dependency treebanks. In S. Dipper and H. Zinsmeister, editors, *Corpus-based Investigations of Pragmatic and Discourse Phenomena*, volume 3 of *Bochumer Linguistische Arbeitsberichte*, pages 83–98. University of Bochum, Bochum, Germany.
- Ivana Kruijff-Korbayova and Mark Steedman. 2003. Discourse and information structure. *Journal of Logic, Language and Information. Special Issue on Discourse and Information Structure*, 12(3):149–259.
- Knud Lambrecht. 1994. *Information Structure and Sentence Form*. Cambridge, U.K.: Cambridge University Press.
- Qing Lu and Lise Getoor. 2003. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning*, Washington, D.C., 21–24 August 2003, pages 496–503.
- Sofus A. Macskassy and Foster Provost. 2007. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983.
- Katja Markert and Malvina Nissim. 2005. Comparing knowledge sources for nominal anaphora resolution. *Computational Linguistics*, 31(3):367–401.
- Josef Meyer and Robert Dale. 2002. Mining a corpus to support associative anaphora resolution. In *Proceedings of the 4th International Conference on Discourse Anaphora and Anaphor Resolution*, Lisbon, Portugal, 18–20 September, 2002.
- Natalia M. Modjeska, Katja Markert, and Malvina Nissim. 2003. Using the web in machine learning for other-anaphora resolution. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, Sapporo, Japan, 11–12 July 2003, pages 176–183.
- Ani Nenkova, Jason Brenier, Anubha Kothari, Sasha Calhoun, Laura Whitton, David Beaver, and Dan Jurafsky. 2007. To memorize or to predict: Prominence labeling in conversational speech. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, N.Y., 22–27 April 2007, pages 9–16.
- Vincent Ng. 2009. Graph-cut-based anaphoricity determination for coreference resolution. In *Proceedings of Human Language Technologies 2009: The Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, Col., 31 May – 5 June 2009, pages 575–583.
- Malvina Nissim, Shipara Dingare, Jean Carletta, and Mark Steedman. 2004. An annotation scheme for information status in dialogue. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal, 26–28 May 2004, pages 1023–1026.

- Malvina Nissim. 2006. Learning information status of discourse entities. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia, 22–23 July 2006, pages 94–102.
- Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, 21–26 July 2004, pages 272–279.
- Massimo Poesio, Rahul Mehta, Axel Maroudas, and Janet Hitzeman. 2004. Learning to resolve bridging references. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, 21–26 July 2004, pages 143–150.
- Massimo Poesio. 2004. The MATE/GNOME proposals for anaphoric annotation, revisited. In *Proceedings of the 5th SIGdial Workshop on Discourse and Dialogue*, Cambridge, Mass., 30 April – 1 May 2004, pages 154–162.
- Scott Prevost. 1996. An information structural approach to spoken language generation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, Cal., 24–27 June 1996, pages 294–301.
- Ellen F. Prince. 1981. Towards a taxonomy of given-new information. In P. Cole, editor, *Radical Pragmatics*, pages 223–255. Academic Press, New York, N.Y.
- Ellen F. Prince. 1992. The ZPG letter: Subjects, definiteness, and information-status. In W.C. Mann and S.A. Thompson, editors, *Discourse Description. Diverse Linguistic Analyses of a Fund-Raising Text*, pages 295–325. John Benjamins, Amsterdam.
- Altaf Rahman and Vincent Ng. 2011. Learning the information status of noun phrases in spoken dialogues. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, U.K., 27–29 July 2011, pages 1069–1080.
- Arndt Rieger, David Lorenz, and Nina Seemann. 2010. A recursive annotation scheme for referential information status. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, La Valetta, Malta, 17–23 May 2010, pages 717–722.
- Julia Ritz, Stefanie Dipper, and Michael Götze. 2008. Annotation of information structure: An evaluation across different types of texts. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, 26 May – 1 June 2008, pages 2137–2142.
- Ryohei Sasano and Sadao Kurohashi. 2009. A probabilistic model for associative anaphora resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, 6–7 August 2009, pages 1455–1464.
- Advaith Siddharthan, Ani Nenkova, and Kathleen McKeown. 2011. Information status distinctions and referring expressions: An empirical study of references to people in news summaries. *Computational Linguistics*, 37(4):811–842.
- Swapna Somasundaran, Galileo Namata, Janyce Wiebe, and Lise Getoor. 2009. Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, 6–7 August 2009.
- Ben Taskar, Pieter Abbeel, and Daphne Koller. 2002. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, Edmonton, Alberta, Canada, 1–4 August 2002, pages 485–492.
- Renata Vieira and Massimo Poesio. 2000. An empirically-based system for processing definite descriptions. *Computational Linguistics*, 26(4):539–593.
- Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2011. OntoNotes release 4.0. LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium.
- Yiming Yang, Seán Slattery, and Rayid Ghani. 2002. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2-3):219–241.
- Guodong Zhou and Fang Kong. 2009. Global learning of noun phrase anaphoricity in coreference resolution via label propagation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, 6–7 August 2009, pages 978–986.



# Structuring E-Commerce Inventory

**Karin Mauge**  
eBay Research Labs  
2145 Hamilton Avenue  
San Jose, CA 95125  
kmauge@ebay.com

**Khash Rohanimanesh**  
eBay Research Labs  
2145 Hamilton Avenue  
San Jose, CA 95125  
krohanimanesh@ebay.com

**Jean-David Ruvini**  
eBay Research Labs  
2145 Hamilton Avenue  
San Jose, CA 95125  
jruvini@ebay.com

## Abstract

Large e-commerce enterprises feature millions of items entered daily by a large variety of sellers. While some sellers provide rich, structured descriptions of their items, a vast majority of them provide unstructured natural language descriptions. In the paper we present a 2 steps method for structuring items into descriptive properties. The first step consists in unsupervised property discovery and extraction. The second step involves supervised property synonym discovery using a maximum entropy based clustering algorithm. We evaluate our method on a year worth of e-commerce data and show that it achieves excellent precision with good recall.

## 1 Introduction

Online commerce has gained a lot of popularity over the past decade. Large on-line C2C marketplaces like eBay and Amazon, feature a very large and long-tail inventory with millions of *items* (product offers) entered into the marketplace every day by a large variety of sellers. While some sellers (generally large professional ones) provide rich, structured description of their products (using schemas or via a global trade item number), the vast majority only provide unstructured natural language descriptions.

To manage items effectively and provide the best user experience, it is critical for these marketplaces to structure their inventory into descriptive name-value pairs (called properties) and ensure that items of the same kind (digital cameras for instance) are described using a unique set of property names

(brand, model, zoom, resolution, etc.) and values. For example, this is important for measuring item similarity and complementarity in merchandising, providing faceted navigation and various business intelligence applications. Note that structuring items does not necessarily mean identifying products as not all e-commerce inventory is manufactured (animals for examples).

Structuring inventory in the e-commerce domain raises several challenges. First, one needs to identify and extract the names and the values used by individual sellers from unstructured textual descriptions. Second, different sellers may describe the same product in very different ways, using different terminologies. For example, Figure 1 shows 3 item descriptions of hard drives from 3 different sellers. The left description mentions "rotational speed" in a specification table while the other two descriptions use the synonym "spindle speed" in a bulleted list (top right) or natural language specifications (bottom right). This requires discovering semantically equivalent property names and values across inventories from multiple sellers. Third, the scale at which on-line marketplaces operate makes impractical to solve any of these problems manually. For instance, eBay reported 99 million active users in 2011, many of whom are sellers, which may translate into thousands or even millions of synonyms to discover across more than 20,000 categories ranging from consumer electronics to collectible and art.

This paper describes a two step process for structuring items in the e-commerce domain. The first step consists in an unsupervised property extraction technique which allows discovering name-value

pairs from unstructured item descriptions. The second step consists in identifying semantically equivalent property names amongst these extracted properties. This is accomplished using supervised maximum entropy based clustering. Note that, although value synonym discovery is an equally important task for structuring items, this is still an area of ongoing research and is not addressed in this paper.

The remainder of this paper is structured as follows. We first review related work. We then describe the two steps of our approach: 1) unsupervised property discovery and extraction and 2) property name synonym discovery. Finally, we present experimental results on real large-scale e-commerce data.

## 2 Related Work

This section reviews related work for the two components of our method, namely unsupervised property extraction and supervised property name synonym discovery.

### 2.1 Unsupervised Property Extraction

A lot of progress has been accomplished in the area of property discovery from product reviews since the pioneering work by (Hu and Liu, 2004). Most of this work is based on the observation, later formalized as *double propagation* by (Qiu et al., 2009), that in reviews, opinion words are usually associated with product properties in some ways, and thus product properties can be identified from opinion words and opinion words from properties alternately and iteratively. While (Hu and Liu, 2004) initially used association mining techniques; (Liu et al., 2005) used Part-Of-Speech and supervised rule mining to generate language patterns and identify product properties; (Popescu and Etzioni, 2005) used point wise mutual information between candidate properties and meronymy discriminators; (Zhuang et al., 2006; Qiu et al., 2009) improved on previous work by using dependency parsing; (Kobayashi et al., 2007) mined property-opinion patterns using statistical and contextual cues; (Wang and Wang, 2008) leveraged property-opinion mutual information and linguistic rules to identify infrequent properties; and (Zhang et al., 2010) proposed a ranking scheme to improve double propagation precision. In this paper, we are focusing on extracting properties from

product descriptions which do not contain opinion words.

In a sense, item properties can be viewed as slots of product templates and our work bears similarities with template induction methods. (Chambers and Jurafsky, 2011) proposed a method for inferring event templates based on word clustering according to their proximity in the corpus and syntactic function clustering. Unfortunately, this technique cannot be applied to our problem due to the lack of discourse redundancy within item descriptions.

(Putthividhya and Hu, 2011) and (Sachan et al., 2011) also addressed the problem of structuring items in the e-commerce domain. However, these works assume that property names are known in advance and focus on discovering values for these properties from very short product titles.

Although we are primarily concerned with unsupervised property discovery, it is worth mentioning (Peng and McCallum, 2004) and (Ghani et al., 2006) who approached the problem using supervised machine learning techniques and require labeled data.

### 2.2 Property Name Synonym Discovery

Our work is related to the synonym discovery research which aims at identifying groups of words that are semantically identical based on some defined similarity metric. The body of work on this problem can be divided into two major approaches (Agirre et al., 2009): methods that are based on the available knowledge resources (e.g., WordNet, or available taxonomies) (Yang and Powers, 2005; Alvarez and Lim, 2007; Hughes and Ramage, ), and methods that use contextual/property distribution around the words (Pereira et al., 1993; Chen et al., 2006; Sahami and Heilman, 2006; Pantel et al., 2009). (Zhai et al., 2010) propose a constrained semi-supervised learning method using a naive Bayes formulation of EM seeded by a small set of labeled data and a set of soft constraints based on the prior knowledge of the problem. There has been also some recent work on applying topic modeling (e.g., LDA) for solving this problem (Guo et al., 2009).

Our work is also related to the existing research on schema matching problem where the objective is to identify objects that are semantically related cross schemas. There has been an extensive study on the


**Ineo I-NA215U+ 640GB Stylish Design Slim Palm-Size SuperSpeed USB 3.0 External Hard Drive** featuring with latest technology SuperSpeed USB 3.0 interface with **transfer rate up to 5Gbps**, (10 times faster than USB 2.0). Expand your PC/Laptop hard drive capacity instantly! SuperSpeed USB 3.0 era has arrived! Imagine you can store and access your data in 5.0Gbps (10 times faster than USB 2.0) with INeo I-NA215U+ External Hard Drive. 5.0Gbps SuperSpeed performance is perfect for ultra speed data backup or video editing and more applications. Also I-NA215U+ is ultra slim, lightweight, stylish design, "highly glossy white" chassis and also for quiet operation.

**Feature**

- **Build-in Capacity: 640GB**
- **SuperSpeed USB 3.0 Interface**
- 10 times faster than USB 2.0
- Fully backwards compatible with USB 2.0/1.1
- USB bus powered, no power adapter required
- No driver required, plug and play!
- Fan-less with silent operation

**Specifications :**

Data Storage	Formatted Capacity	640GB
Data Transfer Rate	To/From Media	USB 3.0 (5.0Gbps) USB 2.0 (480Mbps)
Buffer		8 MB
Seek Time	Average	11 ms
Rotational Speed		5400 RPM
External Interface		USB 3.0 / USB 2.0
Dimensions		5.4(L) x 3.1(W) x 0.5(H)
Warranty		1 Year Warranty



**General Features:**

- 250 GB formatted capacity
- 7200 RPM spindle speed
- 16 MB buffer
- USB 2.0 interface
- Maxtor SafetyDrill software included
- Back up all your files
- Two levels of data security
- Sync data between 2 or more computers
- Customizable Maxtor OneTouch button
- PC and Mac compatible

**Unit Dimensions:**

- 6.75 x 2.5 x 6-inches (H x W x D)
- 2.5 lbs

**Power Specifications:**

- 100-240 VAC, 1.0A, 50-60 VAC, 50-60 Hz; +12V 2.0A
- 24-watt maximum output power

**Description**

The Seagate 500 GB is a slim and portable external hard drive that solves all your data storage problems. Highly energy efficient, this Seagate 2.5-inch HDD consumes less power. The storage capacity of 500GB in this Seagate external HDD stores all your media large files and applications as well. The Seagate 500 GB provides you with an amazing transfer performance from the spindle speed of 5400rpm. The fast USB 2.0 interface of this Seagate 2.5-inch HDD gives you a speedy and reliable performance.

Figure 1: Three examples of item descriptions containing a specification table (left image), a bulleted list (top right) and natural language specifications (bottom right).

problem of schema matching (for a comprehensive survey see (Rahm and Bernstein, 2001; Bellahsene et al., 2011; Bernstein et al., 2011)). In general the work can be classified into rule-based and learning-based approaches. Rule-based systems (Castano and de Antonellis, 1999; Milo and Zohar, 1998; L. Palopoul and Ursino, 1998) often utilize only the schema information (e.g., elements, domain types of schema elements, and schema structure) to define a similarity metric for performing matching among the schema elements in a hard coded fashion. In contrast learning based approaches learn a similarity metric based on both the schema information and the data. Earlier learning based systems (Li and Clifton, 2000; Perkowitz and Etzioni, 1995; Clifton et al., 1997) often rely on one type of learning (e.g., schema meta-data, statistics of the data content, properties of the objects shared between the schemas, etc). These systems do not exploit the complete textual information in the data content therefore have limited applicability. Most recent systems attempt to incorporate the textual contents of the data sources into the system. Doan et

al. (2001) introduce *LSD* which is a semi-automatic machine learning based matching framework that trains a set of base learners using a set of user provided semantic mappings over a small data sources. Each base learner exploits a different type of information, e.g. source schema information and information in the data source. Given a new data source, the base learners are used to discover semantic mappings and their prediction is combined using a meta-learner. Similar to *LSD*, *GLUE* (Doan et al., 2003) also uses a set of base learners combined into a meta-learner for solving the matching problem between two ontologies. Our work is mostly related to (Wick et al., 2008) where they propose a general framework for performing jointly schema matching, co-reference and canonicalization using a supervised machine learning approach. In this approach the matching problem is treated as a clustering problem in the schema attribute space, where a cluster captures a matched set of attributes. A conditional random field (CRF) (Lafferty et al., 2001) is trained using user provided mappings between example schemas, or ontologies. CRF bene-

fits from first order logic features that capture both schema/ontology information as well as textual features in the related data sources.

### 3 Unsupervised Property Extraction

The first step of our solution to structuring e-commerce inventory aims at discovering and extracting relevant properties from items.

Our method is unsupervised and requires no prior knowledge of relevant properties or any domain knowledge as it operates the exact same way for all items and categories. It maintains a set of previously discovered properties called *known properties* with *popularity* information. The popularity of a given property name  $N$  (resp. value  $V$ ) is defined as the number of sellers who are using  $N$  (resp.  $V$ ). A seller is said to use a name or a value if we are able to extract the property name or value from at least one of its item descriptions. The method is incremental in that it starts with an empty set of known properties, mines individual items independently and incrementally builds and updates the set of known properties.

The key intuition is that the abundance of data in e-commerce allows simple and scalable heuristic to perform very well. For property extraction this translates into the following observation: although we may need complex natural language processing for extracting properties from each and every item, simple patterns can extract most of the relevant properties from a subset of the items due to redundancy between sellers. In other words, popular properties are used by many sellers and some of them write their descriptions in a manner that makes these properties easy to extract. For example one pattern that some sellers use to describe product properties often starts by a property name followed by a colon and then the property value (we refer to this pattern as the *colon pattern*). Using this pattern we can mine colon separated short strings like "size : 20 inches" or "color : light blue" which enables us to discover most relevant property names. However, such a pattern extracts properties from a fraction of the inventory only and does not suffice. We are using 4 patterns which are formally defined in Table 1.

All patterns run on the entire item description. Pattern 1 skips the html markers and scripts and

applies only to the content sentences. It ignores any candidate property which name is longer than 30 characters and values longer than 80 characters. These length thresholds may be domain dependent. They have been chosen empirically. Pattern 2, 3 and 4 search for known property names. Pattern 2 extracts the closest value to the right of the name. It allows the name and the value to be separated by special characters or some html markups (like "<TR>", "<TD>", etc.). It captures a wide range of name value pair occurrences including rows of specification tables.

Syntactic cleaning and validation is performed on all the extracted properties. Cleaning consists mainly in removing bullets from the beginning of names and punctuation at the end of names and values. Validation rejects properties which names are pure numbers, properties that contain some special characters and names which are less than 3 characters long. All discovered properties are added to the set of known properties and their popularity counts are updated.

Note that for efficiency reasons, Part-Of-Speech (POS) tagging is performed only on sentences containing the *anchor* of a pattern. The anchor of pattern 1 is the colon sign while the anchor of the other patterns is the known property name KN. We use (Toutanova et al., 2003) for POS tagging.

### 4 Property Synonym Discovery

In this section we briefly overview a probabilistic pairwise property synonym model inspired by (Culotta et al., 2007).

#### 4.1 Probabilistic Model

Given a category  $\mathcal{C}$ , let  $\mathcal{X}_{\mathcal{C}} = \{x_1, x_2, \dots, x_n\}$  be the raw set of  $n$  property names (prior to synonym discovery) extracted from a corpus of data associated with that category. Every property name is associated with pairs of values and popularity count (as defined in Section 3)  $\mathcal{V}_{x_i} = \{(v_j^i, c^i(v_j^i))\}_{j=1}^m$ , where  $v_j^i$  is the  $j^{th}$  value associated for the property name  $x_i$  and  $c^i(v_j^i)$  is the popularity of value  $v_j^i$ . Given a pair of property names  $x_{ij} = \{x_i, x_j\}$ , let the binary random variable  $y_{ij}$  be 1 if  $x_i$  and  $x_j$  are synonyms. Let  $\mathcal{F} = \{f_k(x_{ij}, y)\}$  be a set of features over  $x_{ij}$ . For example,  $f_k(x_{ij}, y)$  may indicate

#	Pattern	Example
1	[NP] [:] [optional DT] [NP]	"color : light blue"
2	[KN] [optional html] [NP]	"size</TD><TD><FONT COLOR="red">20 inches"
3	[!IN] [KN] ["is" or "are"] [NP]	"color is red"
4	[NP] [KN]	"red color"

Table 1: Patterns used to extract properties from item description. The macro tag NP denotes any of the tags NN, NNP, NNS, NNPS, JJ, JJS or CD. The KN tag is defined as a NP tag over a known property name. Pattern 1 only can discover new names; patterns 2 to 4 aim at capturing values for known property names.

whether  $x_i$  and  $x_j$  have both numerical values. Each feature  $f_k$  has an associated real-valued parameter  $\lambda_k$ . The *pairwise* model is given by:

$$\mathcal{P}(y_{ij}|x_{ij}) = \frac{1}{Z_{x_{ij}}} \exp \sum_k \lambda_k f_k(x_{ij}, y_{ij}) \quad (1)$$

where  $Z_{x_{ij}}$  is a normalizer that sums over the two settings of  $y_{ij}$ . This is a maximum-entropy classifier (i.e. logistic regression) in which  $\mathcal{P}(y_{ij}|x_{ij})$  is the probability that  $x_i$  and  $x_j$  are synonyms. To estimate  $\Lambda = \{\lambda_k\}$  from labeled training data, we perform gradient ascent to maximize the log-likelihood of the labeled data.

Given a data set in which property names are manually clustered, the training data can be created by simply enumerating over each pair of synonym property names  $x_{ij}$ , where  $y_{ij}$  is true if  $x_i$  and  $x_j$  are in the same cluster. More practically, given the raw set of extracted properties, first we manually cluster them. Positive examples are then pairs of property names from the same cluster. Negative examples are pairs of names cross two different clusters randomly selected. For example, let assume that the following four property name clusters have been constructed:  $\{color, shade\}$ ,  $\{size, dimension\}$ ,  $\{weight\}$ ,  $\{features\}$ . These clusters implies that "color" and "shade" are synonym; that "size" and "dimension" are synonym and that "weight" and "features" don't have any synonym. The pair  $(color, shade)$  is a positive examples, while  $(size, shade)$  and  $(weight, features)$  are negative examples.

Now, given an unseen category  $\mathcal{C}'$  and the set of raw properties (property names and values) mined from that category, we can construct an undirected-weighted graph in which vertices correspond to the property names  $\mathcal{N}_{\mathcal{C}'}$  and edge weights are propor-

tional to  $\mathcal{P}(y_{ij}|x_{ij})$ . The problem is now reduced to finding the maximum a posteriori (MAP) setting of  $y_{ij}$ s in the new graph. The inference in such models is generally intractable, therefore we apply approximate graph partitioning methods where we partition the graph into clusters with high intra-cluster edge weights and low inter-cluster edge weights. In this work we employ the standard *greedy agglomerative clustering*, in which each noun phrase would be assigned to the most probable cluster according to  $\mathcal{P}(y_{ij}|x_{ij})$ .

## 4.2 Features

Given a pair of property names  $x_{ij} = \{x_i, x_j\}$  we have designed a set of features as follows:

**Property name string similarity/distance:** This measures string similarity between two names. We have included various string edit distances such as *Jaccard* distance over *n-grams* extracted from the property names, and also *Levenstein* distance. We have also included a feature that compares the two property names after their commoner morphological and inflectional endings have been removed using the *Porter Stemmer* algorithm.

**Property value set coverage:** We compute a weighted *Jaccard* measure given the values and the value frequencies associated with a property name.

$$\mathcal{J}(\mathcal{V}_{x_i}, \mathcal{V}_{x_j}) = \frac{\sum_{v \in (\mathcal{V}_{x_i} \cap \mathcal{V}_{x_j})} \min(c^i(v), c^j(v))}{\sum_{v \in (\mathcal{V}_{x_i} \cap \mathcal{V}_{x_j})} \max(c^i(v), c^j(v))}$$

This feature essentially computes how many property values are common between the two property names, weighted by their popularity.

**Property name co-occurrence:** This is an interesting feature which is based on the observation that

two property names that are synonyms, rarely occur together within the same description. This is based on the assumption that sellers are consistent when using property names throughout a single description. For example when they are specifying the size of an item, they either use *size* or *dimensions* exclusively in a single description. However, it is more likely that two property names that are not synonyms appear together within a single description. To conform this assumption, we ran a separate experiment that measures the co-occurrence frequency of the property names in a single category. Table 2 shows a measurement of pairwise co-occurrence of a few example property names computed over the *Audio books* eBay category. Given a property name  $x$  let  $\mathcal{I}(x)$  be the total number of descriptions that contain the name  $x$ . Now, given two property names  $x_i$  and  $x_j$ , we define a measure of co-occurrence of these names as:

$$\text{CO}(x_i, x_j) = \frac{\mathcal{I}(x_i) \cap \mathcal{I}(x_j)}{\mathcal{I}(x_i) \cup \mathcal{I}(x_j)}$$

In Table 2 it can be seen that synonym property names such as "author" and "by" have a zero co-occurrence measure, while semantically different property names such as "format" and "read by" have a non-zero co-occurrence measure.

## 5 Experimental results

This section presents experimental results on a real dataset. We first describe the dataset used for these experiments and then provide results for property extraction and property name synonym discovery.

### 5.1 Data set and methodology

All the results we are reporting in this paper were obtained from a dataset of several billion descriptions corresponding to a year worth of eBay item (no sampling was performed).

For listing an item on eBay, a seller must provide a short descriptive title (up to 80 characters) and can optionally provide a few descriptive name value pairs called item specifics, and a free-form html description. Contrary to item specifics, a vast majority of sellers provide a rich description containing very useful information about the property of their item. Figure 1 shows 3 examples of eBay descriptions.

eBay organizes items into a six-level category structure similar to a topic hierarchy comprising 20,000 leaf categories and covering most of the goods in the world. An item is typically listed in one category but some items may be suitable for and listed in two categories.

Although this dataset is not publicly available, very similar data can be obtained from the eBay web site and through eBay Developers API <sup>1</sup>.

In the following, we report precision and recall results. Evaluation was performed by two annotators (non expert of the domain). For property extraction, they were asked to decide whether or not an extracted property is relevant for the corresponding items; for synonym discovery to decide whether or not sellers refer to the same semantic entity. Annotators were asked to reject the null hypothesis only beyond reasonable doubt and we found the annotator agreement to be extremely high.

### 5.2 Property Extraction Results

We have been running the property extraction method described in Section 3 on our entire dataset. The properties extracted have been aggregated at the leaf category level and ranked by popularity (as defined in Section 3). Because no gold standard data is available for this task, evaluation has to be performed manually. However, it is impractical to review results for 20,000 categories and we uniformly sampled 20 categories randomly.

**Precision.** Table 3 shows the weighted (by category size) average precision of the extracted property names up to rank 20. Precision at rank  $k$  for a given category is defined as the number of relevant properties in the top  $k$  properties of that categories, divided by  $k$ . Table 4 shows the top 15 properties extracted for five eBay categories.

Although we did not formally evaluate the precision of the discovered values, informal reviews have shown that this method extracts good quality values. Examples are "n/a", "well", "storage or well", "would be by well" and "by well" for the property name "Water" in the Land category; "metal", "plastic", "nylon", "acetate" and "durable o matter" for "Frame material" in Sunglasses; or "acrylic",

<sup>1</sup>See <https://www.x.com/developers/eBay/> for details.

	<b>author</b>	<b>by</b>	<b>read by</b>	<b>format</b>	<b>narrated by</b>
<b>author</b>		0	0.06	0.06	0.006
<b>by</b>	0		0.17	0.005	0.013
<b>read by</b>	0.06	0.17		0.035	0
<b>format</b>	0.06	0.005	0.035		0.006
<b>narrated by</b>	0.006	0.013	0	0.006	

Table 2: Co-occurrence measure computed over a subset of property names in the *Audio books* category. Some synonym property names such as *author* and *by* have zero co-occurrence frequency, while semantically different property names such as *format* and *read by* sometimes appear together in some of the item descriptions.

<b>Rank</b>	1	2	3	4	5	6	7	8	9	10
<b>Precision</b>	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.992	0.992	0.986
<b>Rank</b>	11	12	13	14	15	16	17	18	19	20
<b>Precision</b>	0.986	0.997	0.986	1	0.998	1	1	0.959	0.722	0.747

Table 3: Weighted average precision of the top 20 extracted property names.

”oil”, ”acrylic on canvas” and ”oil on canvas” for ”Medium” in Paintings.

Sets of values tend to contain more synonyms than names. Also, we observed that some names exhibit polysemy issues in that their values clearly belong to several semantic clusters. An example of polysemy is the name ”Postmark” in the ”Postcards” categories which contains values like ”none, postally used, no, unused” and years (”1909, 1908, 1910...”). Cleaning and normalizing values is ongoing research effort.

**Recall.** Evaluating recall of our method requires comparing for each category, the number of relevant properties extracted to the number of relevant properties the descriptions in this category contain. It is dauntingly expensive. As a proxy for name recall, we examined 20 categories and found that our method discovered all the relevant popular property names.

It is quite remarkable that an unsupervised method like ours achieves results of that quality and is able to cover most of the good of the world with descriptive properties. To our knowledge, this has never been accomplished before in the e-commerce domain.

### 5.3 Synonym discovery results

To train our name synonym discovery algorithm, we manually clustered properties from 27 randomly se-

lected categories as described in Section 4. This resulted in 178 clusters, 113 of them containing a single property (no synonym) and 65 containing 2 or more properties and capturing actual synonym information. Note that although estimating the co-occurrence table (see Table 2) can be computationally expensive, it is very manageable for such a small set of clusters. Scalability issues due to the large number of eBay categories (nearly 20,000) made impractical to use the solutions proposed in the past to solve that problem as baselines.

Results were produced by applying the trained model to the top 20 discovered properties for each and every eBay categories. The algorithm discovered 10672 synonyms spanning 2957 categories.

**Precision.** To measure the precision of our algorithm, we manually labeled 6618 synonyms as *correct* or *incorrect*. 6076 synonyms were found to be correct and 542 incorrect, a precision of 91.8%. Table 5 shows examples of synonyms and one of the categories where they have been discovered. Some of them are very category specific. For instance, while ”hp” means ”horsepower” for air compressors, it is an acronym of a well known brand in consumer electronics.

**Recall.** Evaluating recall is a more labor intensive task as it involves comparing, for each of the 2957 categories, the number of synonyms discovered to the number of synonyms the category con-

Land	Aquariums	iPod & MP3 Players	Acoustic Guitars	Postcards
State	Dimensions	Weight	Top	Condition
Zoning	Height	Width	Scale length	Publisher
County	Size	Depth	Neck	Size
Water	Width	Height	Bridge	Postmark
Location	Includes	Color	Finish	Postally used
Taxes	Weight	Battery type	Rosette	Type
Size	Depth	Dimensions	Binding	Age
Sewer	Capacity	Frequency response	Fingerboard	Stamp
Power	Color	Storage capacity	Tuning machines	Date
Roads	Power	Display	Case	Title
Lot size	LCD size	Capacity	Pickguard	Postmarked
Utilities	Length	Screen size	Tuners	Subject
Parcel number	Material	Battery	Nut width	Location
	Cable length	Length		Corners
	Condition	Thickness		Era

Table 4: Examples of discovered properties for 5 eBay categories.

Category	Synonyms
Rechargeable Batteries	{Battery type, Chemical composition}
Lodging	{Check-in, Check-in time}
Flower seeds	{Bloom time, Flowering season}
Doors & Door Hardware	{Colour, Color, Main color}
Gemstone	{Cut, Shape}
Air Compressors	{Hp, Horsepower}
Decorative Collectibles	{Item no, Item sku, Item number}
Router Memory	{Memory (ram), Memory size}
Equestrian Clothing	{Bust, Chest}
Traiding Cards	{Rarity, Availability}
Paper Calendar	{Time period, Calendars era}

Table 5: Examples of discovered property name synonyms.

tains. As a proxy we labeled 40 randomly selected categories. For these categories, we found the recall to be 51%. As explained in Section 4, the overlap of values between two names is an important feature for our algorithm. The fact that we are not cleaning and normalizing the values discovered by our property extraction algorithm clearly impacts recall. This is definitively an important direction for further improvements.

## 6 Conclusion

We presented a method for structuring e-commerce inventory into descriptive properties. This method

is based on unsupervised property discovery and extraction from unstructured item descriptions, and on property name synonym discovery achieved using a supervised maximum entropy based clustering algorithm. Experiments on a large real e-commerce dataset showed that both techniques achieve very good results. However, we did not address the issue of property value cleaning and normalization. This is an important direction for future work.



## References

- Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 19–27, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Marco A. Alvarez and SeungJin Lim. 2007. A graph modeling of semantic similarity between words. In *Proceedings of the International Conference on Semantic Computing*, pages 355–362, Washington, DC, USA. IEEE Computer Society.
- Zohra Bellahsene, Angela Bonifati, and Erhard Rahm, editors. 2011. *Schema Matching and Mapping*. Springer.
- Philip A. Bernstein, Jayant Madhavan, and Erhard Rahm. 2011. Generic schema matching, ten years later. *Proceedings of the VLDB Endowment*, 4(11):695–701.
- Silvana Castano and Valeria de Antonellis. 1999. A schema analysis and reconciliation tool environment for heterogeneous databases. In *Proceedings of the 1999 International Symposium on Database Engineering & Applications*, IDEAS '99, pages 53–, Washington, DC, USA. IEEE Computer Society.
- Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 976–986, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Hsin-Hsi Chen, Ming-Shun Lin, and Yu-Chuan Wei. 2006. Novel association measures using web search with double checking. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 1009–1016, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Chris Clifton, Ed Housman, and Arnon Rosenthal. 1997. Experience with a combined approach to attribute-matching across heterogeneous databases. In *In Proc. of the IFIP Working Conference on Data Semantics (DS-7)*.
- Aron Culotta, Michael Wick, Robert Hall, and Andrew McCallum. 2007. First-order probabilistic models for coreference resolution. In *In Proceedings of HLT-NAACL 2007*.
- AnHai Doan, Pedro Domingos, and Alon Y. Halevy. 2001. Reconciling schemas of disparate data sources: a machine-learning approach. In *Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, SIGMOD '01, pages 509–520, New York, NY, USA. ACM.
- AnHai Doan, Jayant Madhavan, Robin Dhamankar, Pedro Domingos, and Alon Halevy. 2003. Learning to match ontologies on the semantic web. *The VLDB Journal*, 12:303–319, November.
- Rayid Ghani, Katharina Probst, Yan Liu, Marko Krema, and Andrew Fano. 2006. Text mining for product attribute extraction. *SIGKDD Explor. Newsl.*, 8:41–48, June.
- Honglei Guo, Huijia Zhu, Zhili Guo, XiaoXun Zhang, and Zhong Su. 2009. Product feature categorization with multilevel latent semantic association. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1087–1096, New York, NY, USA. ACM.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 168–177, New York, NY, USA. ACM.
- Thad Hughes and Daniel Ramage. 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)*, pages 581–589.
- Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.
- D. Saccà L. Palopoli and D. Ursino. 1998. Semi-automatic, semantic discovery of properties from database schemes. In *Proceedings of the 1998 International Symposium on Database Engineering & Applications*, pages 244–, Washington, DC, USA. IEEE Computer Society.
- John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Wen-Syan Li and Chris Clifton. 2000. Semint: a tool for identifying attribute correspondences in heterogeneous databases using neural networks. *Data Knowl. Eng.*, 33:49–84, April.
- Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions

- on the web. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 342–351, New York, NY, USA. ACM.
- Tova Milo and Sagit Zohar. 1998. Using schema matching to simplify heterogeneous data translation. In *Proceedings of the 24rd International Conference on Very Large Data Bases*, VLDB '98, pages 122–133, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2*, EMNLP '09, pages 938–947, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Fuchun Peng and Andrew McCallum. 2004. Accurate information extraction from research papers using conditional random fields. In *HLT-NAACL04*, pages 329–336.
- Fernando Pereira, Naftali Tishby, and Lillian Lee. 1993. Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, ACL '93, pages 183–190, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Mike Perkowitz and Oren Etzioni. 1995. Category translation: learning to understand information on the internet. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, pages 930–936, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 339–346, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Duangmanee Putthividhya and Junling Hu. 2011. Bootstrapped named entity recognition for product attribute extraction. In *EMNLP*, pages 1557–1567.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international joint conference on Artificial intelligence*, IJCAI'09, pages 1199–1204, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Erhard Rahm and Philip A. Bernstein. 2001. A survey of approaches to automatic schema matching. *The VLDB Journal*, 10:334–350.
- Mrinmaya Sachan, Tanveer Faruque, L. V. Subramaniam, and Mukesh Mohania. 2011. Using text reviews for product entity completion. In *Poster at the 5th International Joint Conference on Natural Language Processing*, IJCNLP'11, pages 983–991.
- Mehran Sahami and Timothy 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*, WWW '06, pages 377–386, New York, NY, USA. ACM.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Bo Wang and Houfeng Wang. 2008. Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing. In *Proceedings of the Third International Joint Conference on Natural Language Processing*.
- Michael L. Wick, Khashayar Rohanimanesh, Karl Schultz, and Andrew McCallum. 2008. A unified approach for schema matching, coreference and canonicalization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '08, pages 722–730, New York, NY, USA. ACM.
- Dongqiang Yang and David M. W. Powers. 2005. Measuring semantic similarity in the taxonomy of wordnet. In *Proceedings of the Twenty-eighth Australasian conference on Computer Science - Volume 38*, ACSC '05, pages 315–322, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.
- Zhongwu Zhai, Bing Liu, Hua Xu, and Peifa Jia. 2010. Grouping product features using semi-supervised learning with soft-constraints. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1272–1280, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 1462–1470, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 43–50, New York, NY, USA. ACM.

# Named Entity Disambiguation in Streaming Data

Alexandre Davis<sup>1</sup>, Adriano Veloso<sup>1</sup>, Altigran S. da Silva<sup>2</sup>  
Wagner Meira Jr.<sup>1</sup>, Alberto H. F. Laender<sup>1</sup>

<sup>1</sup>Computer Science Dept. – Federal University of Minas Gerais

<sup>2</sup>Computer Science Dept. – Federal University of Amazonas  
{agdavis, adrianov, meira, laender}@dcc.ufmg.br  
alti@dcc.ufam.edu.br

## Abstract

The named entity disambiguation task is to resolve the many-to-many correspondence between ambiguous names and the unique real-world entity. This task can be modeled as a classification problem, provided that positive and negative examples are available for learning binary classifiers. High-quality sense-annotated data, however, are hard to be obtained in streaming environments, since the training corpus would have to be constantly updated in order to accommodate the fresh data coming on the stream. On the other hand, few positive examples plus large amounts of unlabeled data may be easily acquired. Producing binary classifiers directly from this data, however, leads to poor disambiguation performance. Thus, we propose to enhance the quality of the classifiers using finer-grained variations of the well-known Expectation-Maximization (EM) algorithm. We conducted a systematic evaluation using Twitter streaming data and the results show that our classifiers are extremely effective, providing improvements ranging from 1% to 20%, when compared to the current state-of-the-art biased SVMs, being more than 120 times faster.

## 1 Introduction

Human language is not exact. For instance, an entity<sup>1</sup> may be referred by multiple names (i.e., polysemy), and also the same name may refer to different entities depending on the surrounding context (i.e.,

<sup>1</sup>The term entity refers to anything that has a distinct, separate (materialized or not) existence.

homonymy). The task of named entity disambiguation is to identify which names refer to the same entity in a textual collection (Sarmiento et al., 2009; Yosef et al., 2011; Hoffart et al., 2011). The emergence of new communication technologies, such as micro-blog platforms, brought a humongous amount of textual mentions with ambiguous entity names, raising an urgent need for novel disambiguation approaches and algorithms.

In this paper we address the named entity disambiguation task under a particularly challenging scenario. We are given a stream of messages from a micro-blog channel such as Twitter<sup>2</sup> and a list of names  $n_1, n_2, \dots, n_N$  used for mentioning a specific entity  $e$ . Our problem is to monitor the stream and predict whether an incoming message containing  $n_i$  indeed refers to  $e$  (positive example) or not (negative example). This scenario brings challenges that must be overcome. First, micro-blog messages are composed of a small amount of words and they are written in informal, sometimes cryptic style. These characteristics make hard the identification of entities and the semantics of their relationships (Liu et al., 2011). Further, the scarcity of text in the messages makes it even harder to properly characterize a common context for the entities. Second, as we need to monitor messages that keep coming at a fast pace, we cannot afford to gather information from external sources on-the-fly. Finally, fresh data coming in the stream introduces new patterns, quickly invalidating static disambiguation models.

<sup>2</sup>Twitter is one of the fastest-growing micro-blog channels, and an authoritative source for breaking news (Jansen et al., 2009).

We hypothesize that the lack of information in each individual message and from external sources can be compensated by using information obtained from the large and diverse amount of text in a stream of messages taken as a whole, that is, thousands of messages per second coming from distinct sources.

The information embedded in such a stream of messages may be exploited for entity disambiguation through the application of supervised learning methods, for instance, with the application of binary classifiers. Such methods, however, suffer from a data acquisition bottleneck, since they are based on training datasets that are built by skilled human annotators who manually inspect the messages. This annotation process is usually lengthy and laborious, being clearly unfeasible to be adopted in data streaming scenarios. As an alternative to such manual process, a large amount of unlabeled data, augmented with a small amount of (likely) positive examples, can be collected automatically from the message stream (Liu et al., 2003; Denis, 1998; Comité et al., 1999; Letouzey et al., 2000).

Binary classifiers may be learned from such data by considering unlabeled data as negative examples. This strategy, however, leads to classifiers with poor disambiguation performance, due to a potentially large number of false-negative examples. In this paper we propose to refine binary classifiers iteratively, by performing Expectation-Maximization (EM) approaches (Dempster et al., 1977). Basically, a partial classifier is used to evaluate the likelihood of an unlabeled example being a positive example or a negative example, thus automatically and (continuously) creating a labeled training corpus. This process continues iteratively by changing the label of some examples (an operation we call label-transition), so that, after some iterations, the combination of labels is expected to converge to the one for which the observed data is most likely. Based on such an approach, we introduce novel disambiguation algorithms that differ among themselves on the granularity in which the classifier is updated, and on the label-transition operations that are allowed.

An important feature of the proposed approach is that, at each iteration of the EM-process, a new classifier (an improved one) is produced in order to account for the current set of labeled examples. We introduce a novel strategy to maintain the classifiers

up-to-date incrementally after each iteration, or even after each label-transition operation. Indeed, we theoretically show that our classifier needs to be updated just partially and we are able to determine exactly which parts must be updated, making our disambiguation methods extremely fast.

To evaluate the effectiveness of the proposed algorithms, we performed a systematic set of experiments using large-scale Twitter data containing messages with ambiguous entity names. In order to validate our claims, disambiguation performance is investigated by varying the proportion of false-negative examples in the unlabeled dataset. Our algorithms are compared against a state-of-the-art technique for named entity disambiguation based on classifiers, providing performance gains ranging from 1% to 20% and being roughly 120 times faster.

## 2 Related Work

In the context of databases, traditional entity disambiguation methods rely on similarity functions over attributes associated to the entities (de Carvalho et al., 2012). Obviously, such an approach is unfeasible for the scenario we consider here. Still on databases, Bhattacharya and Getoor (2007) and Dong et. al (2005) propose graph-based disambiguation methods that generate clusters of co-referent entities using known relationships between entities of several types. Methods to disambiguate person names in e-mail (Minkov et al., 2006) and Web pages (Bekkerman and McCallum, 2005; Wan et al., 2005) have employed similar ideas. In e-mails, information taken from the header of the messages leads to establish relationships between users and building a co-reference graph. In Web pages, reference information come naturally from links. Such graph-based approach could hardly be applied to the context we consider, in which the implied relationships between entities mentioned in a given micro-blog message are not clearly defined.

In the case of textual corpora, traditional disambiguation methods represent entity names and their context (Hasegawa et al., 2004) (i.e., words, phrases and other names occurring near them) as weighted vectors (Bagga and Baldwin, 1998; Pedersen et al., 2005). To evaluate whether two names refer to the same entity, these methods compute the similar-

ity between these vectors. Clusters of co-referent names are then built based on such similarity measure. Although effective for the tasks considered in these papers, the simplistic BOW-based approaches they adopt are not suitable for cases in which the context is harder to capture due to the small number of terms available or to informal writing style. To address these problems, some authors argue that contextual information may be enriched with knowledge from external sources, such as search results and the Wikipedia (Cucerzan, 2007; Bunescu and Pasca, 2006; Han and Zhao, 2009). While such a strategy is feasible in an off-line setting, two problems arise when monitoring streams of micro-blog messages. First, gathering information from external sources through the Internet can be costly and, second, informal mentions to named entities make it hard to look for related information in such sources.

The disambiguation methods we propose fall into a learning scenario known as PU (positive and unlabeled) learning (Liu et al., 2003; Denis, 1998; Comité et al., 1999; Letouzey et al., 2000), in which a classifier is built from a set of positive examples plus unlabeled data. Most of the approaches for PU learning, such as the biased-SVM approach (Li and Liu, 2003), are based on extracting negative examples from unlabeled data. We notice that existing approaches for PU learning are not likely to scale given the restrictions imposed by streaming data. Thus, we propose highly incremental approaches, which are able to process large-scale streaming data.

### 3 Disambiguation in Streaming Data

Consider a stream of messages from a micro-blog channel such as Twitter and let  $n_1, n_2, \dots, n_N$  be names used for mentioning a specific entity  $e$  in these messages. Our problem is to continually monitor the stream and predict whether an incoming message containing  $n_i$  indeed refers to  $e$  or not.

This task may be accomplished through the application of classification techniques. In this case, we are given an input data set called the training corpus (denoted as  $\mathcal{D}$ ) which consists of examples of the form  $\langle e, m, c \rangle$ , where  $e$  is the entity,  $m$  is a message containing the entity name (i.e., any  $n_i$ ), and  $c \in \{\ominus, \oplus\}$  is a binary variable that specifies whether or not the entity name in  $m$  refers to the

desired real-world entity  $e$ . The training corpus is used to produce a classifier that relates textual patterns (i.e., terms and sets of terms) in  $m$  to the value of  $c$ . The test set (denoted as  $\mathcal{T}$ ) consists of a set of records of the form  $\langle e, m, ? \rangle$ , and the classifier is used to indicate which messages in  $\mathcal{T}$  refer to (or not) the desired entity.

Supervised classifiers, however, are subject to a data acquisition bottleneck, since the creation of a training corpus requires skilled human annotators to manually inspect the messages. The cost associated with this annotation process may render vast amounts of examples unfeasible. In many cases, however, the acquisition of some positive examples is relatively inexpensive. For instance, as we are dealing with messages collected from micro-blog channels, we may exploit profiles (or hashtags) that are known to be strongly associated with the desired entity. Let us consider, as an illustrative example, the profile associated with a company (i.e., @bayer). Although the entity name is ambiguous, the sense of messages that are posted in this profile is biased towards the entity as being a company. Clearly, other tricks like this one can be used, but, unfortunately, they do not guarantee the absence of false-positives, and they are not complete since the majority of messages mentioning the entity name may appear outside its profile. Thus, the collected examples are not totally reliable, and disambiguation performance would be seriously compromised if classifiers were built upon these uncertain examples directly.

#### 3.1 Expectation-Maximization Approach

In this paper we hypothesize that it is worthwhile to enhance the reliability of unlabeled examples, provided that this type of data is inexpensive and the enhancement effort will be then rewarded with an improvement in disambiguation performance. Thus, we propose a new approach based on the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). We assume two scenarios:

- the training corpus  $\mathcal{D}$  is composed of a small set of *truly* positive examples plus a large amount of unlabeled examples.
- the training corpus  $\mathcal{D}$  is composed of a small set of *potentially* positive examples plus a large amount of unlabeled examples.

In both scenarios, unlabeled examples are initially treated as negative ones, so that classifiers can be built from  $\mathcal{D}$ . Therefore, in both scenarios,  $\mathcal{D}$  may contain false-negatives. In the second scenario, however,  $\mathcal{D}$  may also contain false-positives.

**Definition 3.1:** *The label-transition operation  $x^{\ominus \rightarrow \oplus}$  turns a negative example  $x^{\ominus} \in \mathcal{D}$  into a positive one  $x^{\oplus}$ . The training corpus  $\mathcal{D}$  becomes  $\{(\mathcal{D} - x^{\ominus}) \cup x^{\oplus}\}$ . Similarly, the label-transition operation  $x^{\oplus \rightarrow \ominus}$ , turns a positive example  $x^{\oplus} \in \mathcal{D}$  into a negative one  $x^{\ominus}$ . The training corpus  $\mathcal{D}$  becomes  $\{(\mathcal{D} - x^{\oplus}) \cup x^{\ominus}\}$ .*

Our Expectation Maximization (EM) methods employ a classifier which assigns to each example  $x \in \mathcal{D}$  a probability  $\alpha(x, \ominus)$  of being negative. Then, as illustrated in Algorithm 1, label-transition operations are performed, so that, in the end of the process, it is expected that the assigned labels converge to the combination for which the data is most likely. In the first scenario only operations  $x^{\ominus \rightarrow \oplus}$  are allowed, while in the second scenario operations  $x^{\oplus \rightarrow \ominus}$  are also allowed. In both cases, a crucial issue that affects the effectiveness of our EM-based methods concerns the decision of whether or not performing the label-transition operation. Typically, a transition threshold  $\alpha_{min}$  is employed, so that a label-transition operation  $x^{\ominus \rightarrow \oplus}$  is always performed if  $x$  is a negative example and  $\alpha(x, \ominus) \leq \alpha_{min}$ . Similarly, operation  $x^{\oplus \rightarrow \ominus}$  is always performed if  $x$  is a positive example and  $\alpha(x, \ominus) > \alpha_{min}$ .

---

**Algorithm 1** Expectation-Maximization Approach.

---

**Given:**

$\mathcal{D}$ : training corpus

$\mathcal{R}$ : a binary classifier learned from  $\mathcal{D}$

**Expectation step:**

perform transition operations on examples in  $\mathcal{D}$

**Maximization step:**

update  $\mathcal{R}$  and  $\alpha(x, \ominus) \forall x \in \mathcal{D}$

---

The optimal value for  $\alpha_{min}$  is not known in advance. Fortunately, data distribution may provide hints about proper values for  $\alpha_{min}$ . In our approach, instead of using a single value for  $\alpha_{min}$ , which would be applied to all examples indistinctly, we use a specific  $\alpha_{min}^x$  threshold for each example  $x \in \mathcal{D}$ . Based on such an approach, we in-

troduce fine-grained EM-based methods for named entity disambiguation under streaming data. A specific challenge is that the proposed methods perform several transition operations during each EM iteration, and each transition operation may invalidate parts of the current classifier, which must be properly updated. We take into consideration two possible update granularities:

- the classifier is updated after each EM iteration.
- the classifier is updated after each label-transition operation.

**Incremental Classifier:** As already discussed, the classifier must be constantly updated during the EM process. In this case, well-established classifiers, such as SVMs (Joachims, 2006), have to be learned entirely from scratch, replicating work by large. Thus, we propose as an alternative the use of *Lazy Associative Classifiers* (Velooso et al., 2006).

**Definition 3.2:** *A classification rule is a specialized association rule  $\{\mathcal{X} \rightarrow c\}$  (Agrawal et al., 1993), where the antecedent  $\mathcal{X}$  is a set of terms (i.e., a termset), and the consequent  $c$  indicates if the prediction is positive or negative (i.e.,  $c \in \{\oplus, \ominus\}$ ). The domain for  $\mathcal{X}$  is the vocabulary of  $\mathcal{D}$ . The cardinality of rule  $\{\mathcal{X} \rightarrow c\}$  is given by the number of terms in the antecedent, that is  $|\mathcal{X}|$ . The support of  $\mathcal{X}$  is denoted as  $\sigma(\mathcal{X})$ , and is the number of examples in  $\mathcal{D}$  having  $\mathcal{X}$  as a subset. The confidence of rule  $\{\mathcal{X} \rightarrow c\}$  is denoted as  $\theta(\mathcal{X} \rightarrow c)$ , and is the conditional probability of  $c$  given the termset  $\mathcal{X}$ , that is,  $\theta(\mathcal{X} \rightarrow c) = \frac{\sigma(\mathcal{X} \cup c)}{\sigma(\mathcal{X})}$ .*

In this context, a classifier is denoted as  $\mathcal{R}$ , and it is composed of a set of rules  $\{\mathcal{X} \rightarrow c\}$  extracted from  $\mathcal{D}$ . Specifically,  $\mathcal{R}$  is represented as a pool of entries with the form  $\langle key, properties \rangle$ , where  $key = \{\mathcal{X}, c\}$  and  $properties = \{\sigma(\mathcal{X}), \sigma(\mathcal{X} \cup c), \theta(\mathcal{X} \rightarrow c)\}$ . Each entry in the pool corresponds to a rule, and the key is used to facilitate fast access to rule properties.

Once the classifier  $\mathcal{R}$  is extracted from  $\mathcal{D}$ , rules are collectively used to approximate the likelihood of an arbitrary example being positive ( $\oplus$ ) or negative ( $\ominus$ ). Basically,  $\mathcal{R}$  is interpreted as a poll, in which each rule  $\{\mathcal{X} \rightarrow c\} \in \mathcal{R}$  is a vote given by  $\mathcal{X}$  for  $\oplus$  or  $\ominus$ . Given an example  $x$ , a rule  $\{\mathcal{X} \rightarrow c\}$  is only considered a valid vote if it is applicable to  $x$ .

**Definition 3.3:** A rule  $\{\mathcal{X} \rightarrow c\} \in \mathcal{R}$  is said to be applicable to example  $x$  if  $\mathcal{X} \subseteq x$ , that is, if all terms in  $\mathcal{X}$  are present in example  $x$ .

We denote as  $\mathcal{R}_x$  the set of rules in  $\mathcal{R}$  that are applicable to example  $x$ . Thus, only and all the rules in  $\mathcal{R}_x$  are considered as valid votes when classifying  $x$ . Further, we denote as  $\mathcal{R}_x^c$  the subset of  $\mathcal{R}_x$  containing only rules predicting  $c$ . Votes in  $\mathcal{R}_x^c$  have different weights, depending on the confidence of the corresponding rules. Weighted votes for  $c$  are averaged, giving the score for  $c$  with regard to  $x$  (Equation 1). Finally, the likelihood of  $x$  being a negative example is given by the normalized score (Equation 2).

$$s(x, c) = \sum \frac{\theta(\mathcal{X} \rightarrow c)}{|\mathcal{R}_x^c|}, \text{ with } c \in \{\ominus, \oplus\} \quad (1)$$

$$\alpha(x, \ominus) = \frac{s(x, \ominus)}{s(x, \ominus) + s(x, \oplus)} \quad (2)$$

### Training Projection and Demand-Driven Rule

**Extraction:** Demand-driven rule extraction (Velo et al., 2006) is a recent strategy used to avoid the huge search space for rules, by projecting the training corpus according to the example being processed. More specifically, rule extraction is delayed until an example  $x$  is given for classification. Then, terms in  $x$  are used as a filter that configures the training corpus  $\mathcal{D}$  so that just rules that are applicable to  $x$  can be extracted. This filtering process produces a projected training corpus, denoted as  $\mathcal{D}_x$ , which contains only terms that are present in  $x$ . As shown in (Silva et al., 2011), the number of rules extracted using this strategy grows polynomially with the size of the vocabulary.

**Extending the Classifier Dynamically:** With demand-driven rule extraction, the classifier  $\mathcal{R}$  is extended dynamically as examples are given for classification. Initially  $\mathcal{R}$  is empty; a subset  $\mathcal{R}_{x_i}$  is appended to  $\mathcal{R}$  every time an example  $x_i$  is processed. Thus, after processing a sequence of  $m$  examples  $\{x_1, x_2, \dots, x_m\}$ ,  $\mathcal{R} = \{\mathcal{R}_{x_1} \cup \mathcal{R}_{x_2} \cup \dots \cup \mathcal{R}_{x_m}\}$ .

Before extracting rule  $\{\mathcal{X} \rightarrow c\}$ , it is checked whether this rule is already in  $\mathcal{R}$ . In this case, while processing an example  $x$ , if an entry is found with a key matching  $\{\mathcal{X}, c\}$ , then the rule in  $\mathcal{R}$  is used instead of extracting it from  $\mathcal{D}_x$ . Otherwise, the rule is extracted from  $\mathcal{D}_x$  and then it is inserted into  $\mathcal{R}$ .

**Incremental Updates:** Entries in  $\mathcal{R}$  may become invalid when  $\mathcal{D}$  is modified due to a label-transition operation. Given that  $\mathcal{D}$  has been modified, the classifier  $\mathcal{R}$  must be updated properly. We propose to maintain  $\mathcal{R}$  up-to-date incrementally, so that the updated classifier is exactly the same one that would be obtained by re-constructing it from scratch.

**Lemma 3.1:** Operation  $x^{\ominus \rightarrow \oplus}$  (or  $x^{\oplus \rightarrow \ominus}$ ) does not change the value of  $\sigma(\mathcal{X})$ , for any termset  $\mathcal{X}$ .

**Proof:** The operation  $x^{\ominus \rightarrow \oplus}$  changes only the label associated with  $x$ , but not its terms. Thus, the number of examples in  $\mathcal{D}$  having  $\mathcal{X}$  as a subset is essentially the same as in  $\{(\mathcal{D} - x^{\ominus}) \cup x^{\oplus}\}$ . The same holds for operation  $x^{\oplus \rightarrow \ominus}$ . ■

**Lemma 3.2:** Operation  $x^{\ominus \rightarrow \oplus}$  (or  $x^{\oplus \rightarrow \ominus}$ ) changes the value of  $\sigma(\mathcal{X} \cup c)$  iff termset  $\mathcal{X} \subset x$ .

**Proof:** For operation  $x^{\ominus \rightarrow \oplus}$ , if  $\mathcal{X} \subset x$ , then  $\{\mathcal{X} \cup \ominus\}$  appears once less in  $\{(\mathcal{D} - x^{\ominus}) \cup x^{\oplus}\}$  than in  $\mathcal{D}$ . Similarly,  $\{\mathcal{X} \cup \oplus\}$  appears once more in  $\{(\mathcal{D} - x^{\ominus}) \cup x^{\oplus}\}$  than in  $\mathcal{D}$ . Clearly, if  $\mathcal{X} \not\subset x$ , the number of times  $\{\mathcal{X} \cup \ominus\}$  (and  $\{\mathcal{X} \cup \oplus\}$ ) appears in  $\{(\mathcal{D} - x^{\ominus}) \cup x^{\oplus}\}$  remains the same as in  $\mathcal{D}$ . The same holds for operation  $x^{\oplus \rightarrow \ominus}$ . ■

**Lemma 3.3:** Operation  $x^{\ominus \rightarrow \oplus}$  (or  $x^{\oplus \rightarrow \ominus}$ ) changes the value of  $\theta(\mathcal{X} \rightarrow c)$  iff termset  $\mathcal{X} \subset x$ .

**Proof:** Comes directly from Lemmas 3.1 and 3.2. ■

From Lemmas 3.1 to 3.3, the number of rules that have to be updated due to a label-transition operation is bounded by the number of possible termsets in  $x$ . The following theorem states exactly the rules in  $\mathcal{R}$  that have to be updated due to a transition operation.

**Theorem 3.4:** All rules in  $\mathcal{R}$  that must be updated due to  $x^{\ominus \rightarrow \oplus}$  (or  $x^{\oplus \rightarrow \ominus}$ ) are those in  $\mathcal{R}_x$ .

**Proof:** From Lemma 3.3, all rules  $\{\mathcal{X} \rightarrow c\} \in \mathcal{R}$  that have to be updated due to operation  $x^{\ominus \rightarrow \oplus}$  (or  $x^{\oplus \rightarrow \ominus}$ ) are those for which  $\mathcal{X} \subseteq x$ . By definition,  $\mathcal{R}_x$  contains only and all such rules. ■

Updating  $\theta(\mathcal{X} \rightarrow \ominus)$  and  $\theta(\mathcal{X} \rightarrow \oplus)$  is straightforward. For operation  $x^{\ominus \rightarrow \oplus}$ , it suffices to iterate on  $\mathcal{R}_x$ , incrementing  $\sigma(\mathcal{X} \cup \oplus)$  and decrementing  $\sigma(\mathcal{X} \cup \ominus)$ . Similarly, for operation  $x^{\oplus \rightarrow \ominus}$ , it suffices to iterate on  $\mathcal{R}_x$ , incrementing  $\sigma(\mathcal{X} \cup \ominus)$  and decrementing  $\sigma(\mathcal{X} \cup \oplus)$ . The corresponding values for  $\theta(\mathcal{X} \rightarrow \ominus)$  and  $\theta(\mathcal{X} \rightarrow \oplus)$  are simply obtained by computing  $\frac{\sigma(\mathcal{X} \cup \ominus)}{\sigma(\mathcal{X})}$  and  $\frac{\sigma(\mathcal{X} \cup \oplus)}{\sigma(\mathcal{X})}$ , respectively.

### 3.2 Best Entropy Cut Method

In this section we propose a method for finding the activation threshold,  $\alpha_{min}^x$ , which is a fundamental step of our Expectation-Maximization approach.

**Definition 3.4:** Let  $c^y \in \{\ominus, \oplus\}$  be the label associated with an example  $y \in \mathcal{D}_x$ . Consider  $N_{\ominus}(\mathcal{D}_x)$  the number of examples in  $\mathcal{D}_x$  for which  $c^y = \ominus$ . Similarly, consider  $N_{\oplus}(\mathcal{D}_x)$  the number of examples in  $\mathcal{D}_x$  for which  $c^y = \oplus$ .

**Entropy Minimization:** Our method searches for a threshold  $\alpha_{min}^x$  that provides the best entropy cut in the probability space induced by  $\mathcal{D}_x$ . Specifically, given examples  $\{y_1, y_2, \dots, y_k\}$  in  $\mathcal{D}_x$ , our method first calculates  $\alpha(y_i, \ominus)$  for all  $y_i \in \mathcal{D}_x$ . Then, the values for  $\alpha(y_i, \ominus)$  are sorted in ascending order. In an ideal case, there is a cut  $\alpha_{min}^x$  such that:

$$c^{y_i} = \begin{cases} \oplus & \text{if } \alpha(y_i, \ominus) \leq \alpha_{min}^x \\ \ominus & \text{otherwise} \end{cases}$$

However, there are more difficult cases, for which it is not possible to obtain a perfect separation in the probability space. Thus, we propose a more general method to find the best cut in the probability space. The basic idea is that any value for  $\alpha_{min}^x$  induces two partitions over the space of values for  $\alpha(y_i, \ominus)$  (i.e., one partition with values that are lower than  $\alpha_{min}^x$ , and another partition with values that are higher than  $\alpha_{min}^x$ ). Our method sets  $\alpha_{min}^x$  to the value that minimizes the average entropy of these two partitions. Once  $\alpha_{min}^x$  is calculated, it can be used to activate a label-transition operation. Next we present the basic definitions in order to detail this method.

**Definition 3.5:** Consider a list of pairs  $\mathcal{O} = \{\dots, \langle c^{y_i}, \alpha(y_i, \ominus) \rangle, \langle c^{y_j}, \alpha(y_j, \ominus) \rangle, \dots\}$ , such that  $\alpha(y_i, \ominus) \leq \alpha(y_j, \ominus)$ . Also, consider  $f$  a candidate value for  $\alpha_{min}^x$ . In this case,  $\mathcal{O}_f(\leq)$  is a sub-list of  $\mathcal{O}$ , that is,  $\mathcal{O}_f(\leq) = \{\dots, \langle c^{y_i}, \alpha(y_i, \ominus) \rangle, \dots\}$ , such that for all pairs in  $\mathcal{O}_f(\leq)$ ,  $\alpha(y, \ominus) \leq f$ . Similarly,  $\mathcal{O}_f(>) = \{\dots, \langle c^{y_j}, \alpha(y_j, \ominus) \rangle, \dots\}$ , such that for all pairs in  $\mathcal{O}_f(>)$ ,  $\alpha(y, \ominus) > f$ . In other words,  $\mathcal{O}_f(\leq)$  and  $\mathcal{O}_f(>)$  are partitions of  $\mathcal{O}$  induced by  $f$ .

Our method works as follows. Firstly, it calculates the entropy in  $\mathcal{O}$ , as shown in Equation 3. Then, it calculates the sum of the entropies in each partition induced by  $f$ , according to Equation 4. Finally, it sets  $\alpha_{min}^x$  to the value of  $f$  that minimizes  $E(\mathcal{O}) - E(\mathcal{O}_f)$ , as illustrated in Figure 1.

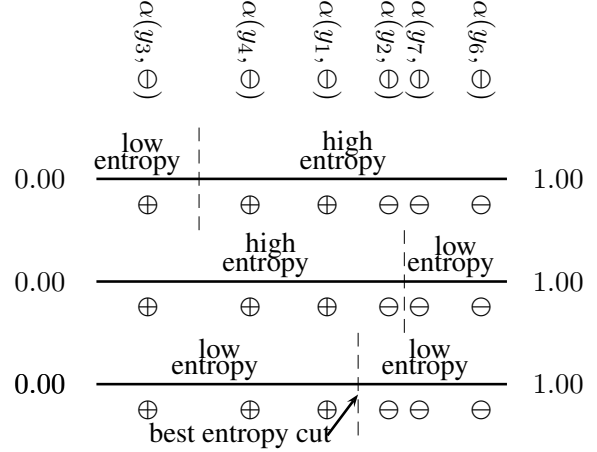


Figure 1: Looking for the minimum entropy cut point.

$$E(\mathcal{O}) = - \left( \frac{N_{\ominus}(\mathcal{O})}{|\mathcal{O}|} \times \log \frac{N_{\ominus}(\mathcal{O})}{|\mathcal{O}|} \right) - \left( \frac{N_{\oplus}(\mathcal{O})}{|\mathcal{O}|} \times \log \frac{N_{\oplus}(\mathcal{O})}{|\mathcal{O}|} \right) \quad (3)$$

$$E(\mathcal{O}_f) = \frac{|\mathcal{O}_f(\leq)|}{|\mathcal{O}|} \times E(\mathcal{O}_f(\leq)) + \frac{|\mathcal{O}_f(>)|}{|\mathcal{O}|} \times E(\mathcal{O}_f(>)) \quad (4)$$

### 3.3 Disambiguation Algorithms

In this section we discuss four algorithms based on our incremental EM approach and following our Best Entropy Cut method. They differ among themselves on the granularity in which the classifier is updated and on the label-transition operations allowed:

- A1: the classifier is updated incrementally after each EM iteration (which may comprise several label-transition operations). Only operation  $x^{\ominus \rightarrow \oplus}$  is allowed.
- A2: the classifier is updated incrementally after each EM iteration. Both operations  $x^{\ominus \rightarrow \oplus}$  and  $x^{\oplus \rightarrow \ominus}$  are allowed.
- A3: the classifier is updated incrementally after each label-transition operation. Only operation  $x^{\ominus \rightarrow \oplus}$  is allowed.
- A4: the classifier is updated incrementally after each label-transition operation. Both operations  $x^{\ominus \rightarrow \oplus}$  and  $x^{\oplus \rightarrow \ominus}$  are allowed.



## 4 Experimental Evaluation

In this section we analyze our algorithms using standard measures such as AUC values. For each positive+unlabeled (PU) corpus used in our evaluation we randomly selected  $x\%$  of the positive examples (P) to become unlabeled ones (U). This procedure enables us to control the uncertainty level of the corpus. For each level we have a different TPR-FPR combination, enabling us to draw ROC curves. We repeated this procedure five times, so that five executions were performed for each uncertainty level. Tables 2–5 show the average for the five runs. Wilcoxon significance tests were performed ( $p < 0.05$ ) and best results, including statistical ties, are shown in bold.

### 4.1 Baselines and Collections

Our baselines include namely SVMs (Joachims, 2006) and Biased SVMs (B-SVM (Liu et al., 2003)). Although the simple SVM algorithm does not adapt itself with unlabeled data, we decided to use it in order to get a sense of the performance achieved by simple baselines (in this case, unlabeled data is simply used as negative examples). The B-SVM algorithm uses a soft-margin SVM as the underlying classifier, which is re-constructed from scratch after each EM iteration. B-SVM employs a single transition threshold  $\alpha_{min}$  for the entire corpus, instead of a different threshold  $\alpha_{min}^x$  for each  $x \in \mathcal{D}$ . It is representative of the state-of-the-art for learning classifiers from PU data.

We employed two different Twitter collections. The first collection, ORGANIZATIONS, is composed of 10 corpora<sup>3</sup> ( $O_1$  to  $O_{10}$ ). Each corpus contains messages in English mentioning the name of an organization (Bayer, Renault, among others). All messages were labeled by five annotators. Label  $\oplus$  means that the message is associated with the organization, whereas label  $\ominus$  means the opposite.

The other collection, SOCCER TEAMS, contains 6 large-scale PU corpora ( $ST_1$  to  $ST_6$ ), taken from a platform for real time event monitoring (the link to this platform is omitted due to blind review). Each corpus contains messages in Portuguese mentioning the name/mascot of a Brazilian soccer team. Both collections are summarized in Table 1.

<sup>3</sup><http://nlp.uned.es/weps/>

Table 1: Characteristics of each collection.

	P	U		P	U
$O_1$	404	10	$ST_1$	216,991	251,198
$O_2$	404	55	$ST_2$	256,027	504,428
$O_3$	349	116	$ST_3$	160,706	509,670
$O_4$	329	119	$ST_4$	147,706	633,357
$O_5$	335	133	$ST_5$	35,021	168,669
$O_6$	314	143	$ST_6$	5,993	351,882
$O_7$	292	148	–	–	–
$O_8$	295	172	–	–	–
$O_9$	273	165	–	–	–
$O_{10}$	33	425	–	–	–

### 4.2 Results

All experiments were performed on a Linux PC with an Intel Core 2 Duo 2.20GHz and 4GBytes RAM. Next we discuss the disambiguation performance and the computational efficiency of our algorithms.

**ORGANIZATIONS Corpora:** Table 2 shows average AUC values for each algorithm. Algorithm A4 was the best performer in all cases, suggesting the benefits of (i) enabling both types of label-transition operations and (ii) keeping the classifier up-to-date after each label-transition operation. Further, algorithm A3 performed better than algorithm A2 in most of the cases, indicating the importance of keeping the classifier always up-to-date. On average A1 provides gains of 4% when compared against B-SVM, while A4 provides gains of more than 20%.

**SOCCER TEAMS Corpora:** Table 3 shows average AUC values for each algorithm. Again, algorithm A4 was the best performer, providing gains that are up to 13% when compared against the baseline. Also, algorithm A3 performed better than algorithm A2, and the effectiveness of Algorithm A1 is similar to the effectiveness of the baseline.

Since the SOCCER TEAMS collection is composed of large-scale corpora, in addition to high effectiveness, another important issue to be evaluated is computational performance. Table 4 shows the results obtained for the evaluation of our algorithms. As it can be seen, algorithm A1 is the fastest one, since it is the simplest one. Even though being slower than algorithm A1, algorithm A4 runs, on average, 120 times faster than B-SVM.

Table 2: Average AUC values for each algorithm.

	A1	A2	A3	A4	SVM	B-SVM
O <sub>1</sub>	0.74 ± 0.02	0.76 ± 0.02	0.74 ± 0.03	<b>0.79</b> ± 0.01	0.71 ± 0.03	0.76 ± 0.01
O <sub>2</sub>	0.77 ± 0.02	0.78 ± 0.02	0.70 ± 0.03	<b>0.82</b> ± 0.02	0.73 ± 0.03	0.75 ± 0.02
O <sub>3</sub>	<b>0.68</b> ± 0.02	<b>0.70</b> ± 0.01	<b>0.69</b> ± 0.02	<b>0.69</b> ± 0.02	0.64 ± 0.03	0.65 ± 0.02
O <sub>4</sub>	0.68 ± 0.02	0.68 ± 0.02	<b>0.70</b> ± 0.01	<b>0.72</b> ± 0.02	0.63 ± 0.02	0.66 ± 0.02
O <sub>5</sub>	<b>0.71</b> ± 0.01	<b>0.72</b> ± 0.01	<b>0.71</b> ± 0.01	<b>0.72</b> ± 0.01	0.69 ± 0.01	<b>0.71</b> ± 0.01
O <sub>6</sub>	0.73 ± 0.01	0.73 ± 0.01	<b>0.75</b> ± 0.01	<b>0.75</b> ± 0.01	0.68 ± 0.02	0.70 ± 0.01
O <sub>7</sub>	0.69 ± 0.01	0.72 ± 0.01	<b>0.74</b> ± 0.01	<b>0.74</b> ± 0.01	0.66 ± 0.02	0.69 ± 0.02
O <sub>8</sub>	0.65 ± 0.02	0.68 ± 0.02	0.69 ± 0.02	<b>0.72</b> ± 0.01	0.61 ± 0.03	0.63 ± 0.03
O <sub>9</sub>	0.70 ± 0.01	0.70 ± 0.01	<b>0.72</b> ± 0.01	<b>0.72</b> ± 0.01	0.65 ± 0.01	0.70 ± 0.01
O <sub>10</sub>	0.70 ± 0.01	<b>0.74</b> ± 0.02	0.71 ± 0.02	<b>0.75</b> ± 0.02	0.61 ± 0.03	0.66 ± 0.02

Table 3: Average AUC values for each algorithm.

	A1	A2	A3	A4	SVM	B-SVM
ST <sub>1</sub>	0.62 ± 0.02	0.63 ± 0.02	0.64 ± 0.01	<b>0.67</b> ± 0.02	0.59 ± 0.03	0.61 ± 0.03
ST <sub>2</sub>	0.55 ± 0.01	<b>0.58</b> ± 0.01	<b>0.59</b> ± 0.01	<b>0.59</b> ± 0.01	0.54 ± 0.01	0.57 ± 0.01
ST <sub>3</sub>	0.65 ± 0.02	0.67 ± 0.01	0.67 ± 0.01	<b>0.69</b> ± 0.01	0.61 ± 0.03	0.64 ± 0.03
ST <sub>4</sub>	0.57 ± 0.01	<b>0.59</b> ± 0.01	<b>0.59</b> ± 0.01	<b>0.59</b> ± 0.01	0.50 ± 0.04	0.55 ± 0.02
ST <sub>5</sub>	0.74 ± 0.01	0.74 ± 0.01	<b>0.77</b> ± 0.02	<b>0.77</b> ± 0.01	0.67 ± 0.02	0.72 ± 0.03
ST <sub>6</sub>	0.68 ± 0.02	0.70 ± 0.01	<b>0.71</b> ± 0.01	<b>0.72</b> ± 0.01	0.63 ± 0.01	0.68 ± 0.02

Table 4: Average execution time (secs) for each algorithm. The time spent by algorithm A1 is similar to the time spent by algorithm A2. The time spent by algorithm A3 is similar to the time spent by algorithm A4.

	A1(≈A2)	A3(≈A4)	SVM	B-SVM
ST <sub>1</sub>	1,565	2,102	9,172	268,216
ST <sub>2</sub>	2,086	2,488	11,284	297,556
ST <sub>3</sub>	2,738	3,083	14,917	388,184
ST <sub>4</sub>	847	1,199	6,188	139,100
ST <sub>5</sub>	1,304	1,604	9,017	192,576
ST <sub>6</sub>	1,369	1,658	9,829	196,922

## 5 Conclusions

In this paper we have introduced a novel EM approach, which employs a highly incremental underlying classifier based on association rules, completely avoiding work replication. Further, two label-transition operations are allowed, enabling the correction of false-negatives and false-positives. We proposed four algorithms based on our EM approach. Our algorithms employ an entropy min-

imization method, which finds the best transition threshold for each example in  $\mathcal{D}$ . All these properties make our algorithms appropriate for named entity disambiguation under streaming data scenarios. Our experiments involve Twitter data mentioning ambiguous named entities. These datasets were obtained from real application scenarios and from platforms currently in operation. We have shown that three of our algorithms achieve significantly higher disambiguation performance when compared against a strong baseline (B-SVM), providing gains ranging from 1% to 20%. Also importantly, for large-scale streaming data, our algorithms are more than 120 times faster than the baseline.

## 6 Acknowledgments

This research is supported by InWeb – The Brazilian National Institute of Science and Technology for the Web (CNPq grant no. 573871/2008-6), by UOL (www.uol.com.br) through its UOL Bolsa Pesquisa program (process number 20110215172500), and by the authors’ individual grants from CAPES, CNPq and Fapemig.

## References

- R. Agrawal, T. Imielinski and A. Swami. 1993. Mining association rules between sets of items in large databases. In *Proceedings of the 18th ACM SIGMOD International Conference on Management of Data, Washington, D.C.*, pages 207–216.
- A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 17th International Conference on Computational Linguistics, Montreal, Canada*, pages 79–85.
- R. Bekkerman and A. McCallum. 2005. Disambiguating web appearances of people in a social network. In *Proceedings of the 14th International Conference on the World Wide Web, Chiba, Japan*, pages 463–470.
- I. Bhattacharya and L. Getoor. 2007. Collective entity resolution in relational data. *ACM Transactions on Knowledge Discovery from Data*, 1.
- R. Bunescu and M. Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11st Conference of the European Chapter of the Association for Computational Linguistics, Proceedings of the Conference, Trento, Italy*, pages 9–16.
- F. De Comit e, F. Denis, R. Gilleron and F. Letouzey. 1999. Positive and unlabeled examples help learning. In *Proceedings of the 10th International Conference on Algorithmic Learning Theory, Tokyo, Japan*, pages 219–230.
- S. Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of the 4th Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic*, pages 708–716.
- M. G. de Carvalho, A. H. F. Laender, M. A. Gonalves, and A. S. da Silva. 2006. Learning to deduplicate. *Proceedings of the 6th ACM/IEEE Joint Conference on Digital Libraries, Chapel Hill, NC, USA*. pages 41–50.
- A. Dempster, N. Laird, and D. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.
- F. Denis. 1998. PAC learning from positive statistical queries. In *Proceedings of the Algorithmic Learning Theory, 9th International Conference, Otzenhausen, Germany*, pages 112–126.
- X. Dong, A. Y. Halevy, and J. Madhavan. 2005. Reference reconciliation in complex information spaces. In *Proceedings of the 24th ACM SIGMOD International Conference on Management of Data, Baltimore, USA*, pages 85–96.
- X. Han and J. Zhao. 2009. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceedings of the 18th ACM conference on Information and knowledge management, Hong Kong, China*, pages 215–224.
- T. Hasegawa, S. Sekine and R. Grishman. 2004. Discovering Relations among Named Entities from Large Corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain*, pages 415–422.
- J. Hoffart, M. Yosef, I. Bordino, H. F urstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater and G. Weikum. 2011. Robust Disambiguation of Named Entities in Text. In *Proceedings of the 8th Conference on Empirical Methods in Natural Language Processing, Edinburgh, UK*, pages 782–792.
- B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. 2009. Twitter power: Tweets as electronic word of mouth. *JASIST*, 60(11):2169–2188.
- T. Joachims. 2006. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, USA*, pages 217–226.
- F. Letouzey, F. Denis, and R. Gilleron. 2000. Learning from positive and unlabeled examples. In *Proceedings of the 11th International Conference on Algorithmic Learning Theory, Sydney, Australia*, pages 71–85.
- X. Li and B. Liu. 2003. Learning to classify texts using positive and unlabeled data. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence, Acapulco, Mexico*, pages 587–592.
- B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu. 2003. Building text classifiers using positive and unlabeled examples. In *Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, USA*, pages 179–188.
- X. Liu, S. Zhang, F. Wei and M. Zhou 2011. Recognizing Named Entities in Tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA*, pages 359–367.
- E. Minkov, W. W. Cohen, and A. Y. Ng. 2006. Contextual search and name disambiguation in email using graphs. In *Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, USA*, pages 27–34.
- T. Pedersen, A. Purandare, and A. Kulkarni. 2005. Name discrimination by clustering similar contexts. In *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing, Mexico City, Mexico*, pages 226–237.
- I. S. Silva, J. Gomide, A. Veloso, W. Meira Jr. and R. Ferreira 2011. Effective sentiment stream analysis with

- self-augmenting training and demand-driven projection. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, Beijing, China*, pages 475–484.
- L. Sarmiento, A. Kehlenbeck, E. Oliveira, and L. Ungar. 2009. An approach to web-scale named-entity disambiguation. In *Proceedings of the 6th International Conference on Machine Learning and Data Mining in Pattern Recognition, Leipzig, Germany*, pages 689–703.
- A. Veloso, W. Meira Jr., M. de Carvalho, B. Pôssas, S. Parthasarathy, and M. J. Zaki. 2002. Mining frequent itemsets in evolving databases. In *Proceedings of the Second SIAM International Conference on Data Mining, Arlington, USA*.
- A. Veloso, W. Meira Jr., and M. J. Zaki. 2006. Lazy associative classification. In *Proceedings of the 6th IEEE International Conference on Data Mining, Hong Kong, China*, pages 645–654.
- X. Wan, J. Gao, M. Li, and B. Ding. 2005. Person resolution in person search results: Webhawk. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management, Bremen, Germany*, pages 163–170.
- M. Yosef, J. Hoffart, I. Bordino, M. Spaniol and G. Weikum. 2011. AIDA: An Online Tool for Accurate Disambiguation of Named Entities in Text and Tables. *PVLDB*, 4(12):1450–1453.

# Big Data versus the Crowd: Looking for Relationships in All the Right Places

Ce Zhang    Feng Niu    Christopher Ré    Jude Shavlik

Department of Computer Sciences

University of Wisconsin-Madison, USA

{czhang, leonn, chrisre, shavlik}@cs.wisc.edu

## Abstract

Classically, training relation extractors relies on high-quality, manually annotated training data, which can be expensive to obtain. To mitigate this cost, NLU researchers have considered two newly available sources of less expensive (but potentially lower quality) labeled data from distant supervision and crowd sourcing. There is, however, no study comparing the relative impact of these two sources on the precision and recall of post-learning answers. To fill this gap, we empirically study how state-of-the-art techniques are affected by scaling these two sources. We use corpus sizes of up to 100 million documents and tens of thousands of crowd-source labeled examples. Our experiments show that increasing the corpus size for distant supervision has a statistically significant, positive impact on quality (F1 score). In contrast, human feedback has a positive and statistically significant, but lower, impact on precision and recall.

## 1 Introduction

Relation extraction is the problem of populating a *target relation* (representing an entity-level relationship or attribute) with facts extracted from natural-language text. Sample relations include people’s titles, birth places, and marriage relationships.

Traditional relation-extraction systems rely on manual annotations or domain-specific rules provided by experts, both of which are scarce resources that are not portable across domains. To remedy these problems, recent years have seen interest in the *distant supervision* approach for rela-

tion extraction (Wu and Weld, 2007; Mintz et al., 2009). The input to distant supervision is a set of *seed facts* for the target relation together with an (unlabeled) text corpus, and the output is a set of (noisy) annotations that can be used by any machine learning technique to train a statistical model for the target relation. For example, given the target relation `birthPlace(person, place)` and a seed fact `birthPlace(John, Springfield)`, the sentence “*John and his wife were born in Springfield in 1946*” (S1) would qualify as a positive training example.

Distant supervision replaces the expensive process of manually acquiring annotations that is required by direct supervision with resources that already exist in many scenarios (seed facts and a text corpus). On the other hand, distantly labeled data may not be as accurate as manual annotations. For example, “*John left Springfield when he was 16*” (S2) would also be considered a positive example about place of birth by distant supervision as it contains both John and Springfield. The hypothesis is that the broad coverage and high redundancy in a large corpus would compensate for this noise. For example, with a large enough corpus, a distant supervision system may find that patterns in the sentence S1 strongly correlate with seed facts of `birthPlace` whereas patterns in S2 do not qualify as a strong indicator. Thus, intuitively the quality of distant supervision should improve as we use larger corpora. However, there has been no study on the impact of corpus size on distant supervision for relation extraction. Our goal is to fill this gap.

Besides “big data,” another resource that may be valuable to distant supervision is crowdsourc-

ing. For example, one could employ crowd workers to provide feedback on whether distant supervision examples are correct or not (Gormley et al., 2010). Intuitively the crowd workforce is a perfect fit for such tasks since many erroneous distant labels could be easily identified and corrected by humans. For example, distant supervision may mistakenly consider “*Obama took a vacation in Hawaii*” a positive example for `birthPlace` simply because a database says that Obama was born in Hawaii; a crowd worker would correctly point out that this sentence is not actually indicative of this relation.

It is unclear however which strategy one should use: scaling the text corpus or the amount of human feedback. Our primary contribution is to empirically assess how scaling these inputs to distant supervision impacts its result quality. We study this question with input data sets that are orders of magnitude larger than those in prior work. While the largest corpus (Wikipedia and New York Times) employed by recent work on distant supervision (Mintz et al., 2009; Yao et al., 2010; Hoffmann et al., 2011) contain about 2M documents, we run experiments on a 100M-document (50X more) corpus drawn from ClueWeb.<sup>1</sup> While prior work (Gormley et al., 2010) on crowdsourcing for distant supervision used thousands of human feedback units, we acquire tens of thousands of human-provided labels. Despite the large scale, we follow state-of-the-art distant supervision approaches and use deep linguistic features, e.g., part-of-speech tags and dependency parsing.<sup>2</sup>

Our experiments shed insight on the following two questions:

1. *How does increasing the corpus size impact the quality of distant supervision?*
2. *For a given corpus size, how does increasing the amount of human feedback impact the quality of distant supervision?*

We found that increasing corpus size consistently and significantly improves recall and F1, despite reducing precision on small corpora; in contrast, human feedback has relatively small impact on precision and recall. For example, on a TAC corpus with 1.8M documents, we found that increasing the corpus size ten-fold consistently results in statistically

significant improvement in F1 on two standardized relation extraction metrics (t-test with  $p=0.05$ ). On the other hand, increasing human feedback amount ten-fold results in statistically significant improvement on F1 only when the corpus contains at least 1M documents; and the magnitude of such improvement was only one fifth compared to the impact of corpus-size increment.

We find that the quality of distant supervision tends to be *recall gated*, that is, for any given relation, distant supervision fails to find all possible linguistic signals that indicate a relation. By expanding the corpus one can expand the number of patterns that occur with a known set of entities. Thus, as a rule of thumb for developing distant supervision systems, one should first attempt to expand the training corpus and then worry about precision of labels only after having obtained a broad-coverage corpus.

Throughout this paper, it is important to understand the difference between *mentions* and *entities*. Entities are conceptual objects that exist in the world (e.g., Barack Obama), whereas authors use a variety of wordings to refer to (which we call “mention”) entities in text (Ji et al., 2010).

## 2 Related Work

The idea of using entity-level structured data (e.g., facts in a database) to generate mention-level training data (e.g., in English text) is a classic one: researchers have used variants of this idea to extract entities of a certain type from webpages (Hearst, 1992; Brin, 1999). More closely related to relation extraction is the work of Lin and Patel (2001) that uses dependency paths to find answers that express the same relation as in a question.

Since Mintz et al. (2009) coined the name “*distant supervision*,” there has been growing interest in this technique. For example, distant supervision has been used for the TAC-KBP slot-filling tasks (Surdeanu et al., 2010) and other relation-extraction tasks (Hoffmann et al., 2010; Carlson et al., 2010; Nguyen and Moschitti, 2011a; Nguyen and Moschitti, 2011b). In contrast, we study how increasing input size (and incorporating human feedback) improves the result quality of distant supervision.

We focus on logistic regression, but it is interesting future work to study more sophisticated prob-

<sup>1</sup><http://lemurproject.org/clueweb09.php/>

<sup>2</sup>We used 100K CPU hours to run such tools on ClueWeb.

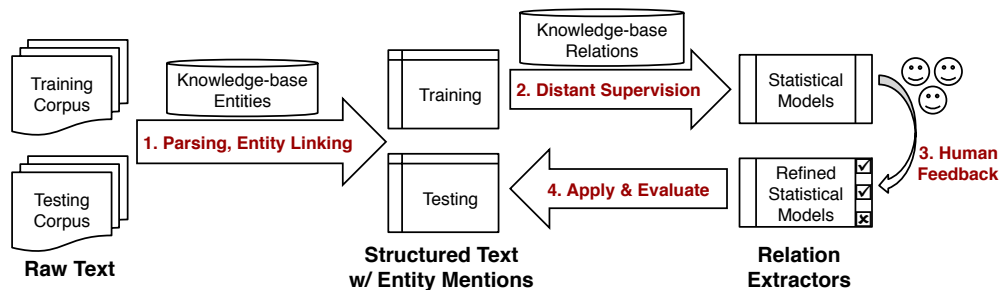


Figure 1: The workflow of our distant supervision system. Step 1 is preprocessing; step 4 is final evaluation. The key steps are distant supervision (step 2), where we train a logistic regression (LR) classifier for each relation using (noisy) examples obtained from sentences that match Freebase facts, and human feedback (step 3) where a crowd workforce refines the LR classifiers by providing feedback to the training data.

abilistic models; such models have recently been used to relax various assumptions of distant supervision (Riedel et al., 2010; Yao et al., 2010; Hoffmann et al., 2011). Specifically, they address the noisy assumption that, if two entities participate in a relation in a knowledge base, then all co-occurrences of these entities express this relation. In contrast, we explore the effectiveness of increasing the training data sizes to improve distant-supervision quality.

Sheng et al. (2008) and Gormley et al. (2010) study the quality-control issue for collecting training labels via crowdsourcing. Their focus is the collection process; in contrast, our goal is to quantify the impact of this additional data source on distant-supervision quality. Moreover, we experiment with one order of magnitude more human labels. Hoffmann et al. (2009) study how to acquire end-user feedback on relation-extraction results posted on an augmented Wikipedia site; it is interesting future work to integrate this source in our experiments. One technique for obtaining human input is active learning. We tried several active-learning techniques as described by Settles (2010), but did not observe any notable advantage over uniform sampling-based example selection.<sup>3</sup>

### 3 Distant Supervision Methodology

Relation extraction is the task of identifying relationships between mentions, in natural-language text, of entities. An example relation is that two persons are married, which for mentions of entities  $x$  and  $y$  is denoted  $R(x, y)$ . Given a corpus  $C$  con-

taining mentions of named entities, our goal is to learn a classifier for  $R(x, y)$  using linguistic features of  $x$  and  $y$ , e.g., dependency-path information. The problem is that we lack the large amount of labeled examples that are typically required to apply supervised learning techniques. We describe an overview of these techniques and the methodological choices we made to implement our study. Figure 1 illustrates the overall workflow of a distant supervision system. At each step of the distant supervision process, we closely follow the recent literature (Mintz et al., 2009; Yao et al., 2010).

#### 3.1 Distant Supervision

Distant supervision compensates for a lack of training examples by generating what are known as *silver-standard examples* (Wu and Weld, 2007). The observation is that we are often able to obtain a structured, but incomplete, database  $D$  that instantiates relations of interest and a text corpus  $C$  that contains mentions of the entities in our database. Formally, a database is a tuple  $D = (E, \bar{R})$  where  $E$  is a set of entities and  $\bar{R} = (R_1 \dots, R_N)$  is a tuple of instantiated predicates. For example,  $R_i$  may contain pairs of married people.<sup>4</sup> We use the facts in  $R_i$  combined with  $C$  to generate examples.

Following recent work (Mintz et al., 2009; Yao et al., 2010; Hoffmann et al., 2011), we use Freebase<sup>5</sup> as the knowledge base for seed facts. We use two text corpora: (1) the TAC-KBP<sup>6</sup> 2010 corpus that

<sup>3</sup>More details in our technical report (Zhang et al., 2012).

<sup>4</sup>We only consider binary predicates in this work.

<sup>5</sup><http://freebase.com>

<sup>6</sup>KBP stands for “Knowledge-Base Population.”

consists of 1.8M newswire and blog articles<sup>7</sup>, and (2) the ClueWeb09 corpus that is a 2009 snapshot of 500M webpages. We use the TAC-KBP slot filling task and select those TAC-KBP relations that are present in the Freebase schema as targets (20 relations on people and organization).

One problem is that relations in  $D$  are defined at the entity level. Thus, the pairs in such relations are not embedded in text, and so these pairs lack the linguistic context that we need to extract features, i.e., the features used to describe examples. In turn, this implies that these pairs cannot be used directly as training examples for our classifier. To generate training examples, we need to map the entities back to mentions in the corpus. We denote the relation that describes this mapping as the relation  $EL(e, m)$  where  $e \in E$  is an entity in the database  $D$  and  $m$  is a mention in the corpus  $C$ . For each relation  $R_i$ , we generate a set of (noisy) positive examples denoted  $R_i^+$  defined as  $R_i^+ =$

$$\{(m_1, m_2) \mid R(e_1, e_2) \wedge EL(e_1, m_1) \wedge EL(e_2, m_2)\}$$

As in previous work, we impose the constraint that both mentions  $(m_1, m_2) \in R_i^+$  are contained in the same sentence (Mintz et al., 2009; Yao et al., 2010; Hoffmann et al., 2011). To generate negative examples for each relation, we follow the assumption in Mintz et al. (2009) that relations are disjoint and sample from other relations, i.e.,  $R_i^- = \cup_{j \neq i} R_j^+$ .

### 3.2 Feature Extraction

Once we have constructed the set of possible mention pairs, the state-of-the-art technique to generate feature vectors uses linguistic tools such as part-of-speech taggers, named-entity recognizers, dependency parsers, and string features. Following recent work on distant supervision (Mintz et al., 2009; Yao et al., 2010; Hoffmann et al., 2011), we use both lexical and syntactic features. After this stage, we have a well-defined machine learning problem that is solvable using standard supervised techniques. We use *sparse logistic regression* ( $\ell_1$  regularized) (Tibshirani, 1996), which is used in previous studies. Our feature extraction process consists of three steps:

1. Run Stanford CoreNLP with POS tagging and named entity recognition (Finkel et al., 2005);
2. Run dependency parsing on TAC with the Ensemble parser (Surdeanu and Manning, 2010) and on ClueWeb with MaltParser (Nivre et al., 2007)<sup>8</sup>; and
3. Run a simple entity-linking system that utilizes NER results and string matching to identify mentions of Freebase entities (with types).<sup>9</sup>

The output of this processing is a repository of structured objects (with POS tags, dependency parse, and entity types and mentions) for sentences from the training corpus. Specifically, for each pair of entity mentions  $(m_1, m_2)$  in a sentence, we extract the following features  $F(m_1, m_2)$ : (1) the word sequence (including POS tags) between these mentions after normalizing entity mentions (e.g., replacing “John Nolen” with a place holder PER); if the sequence is longer than 6, we take the 3-word prefix and the 3-word suffix; (2) the dependency path between the mention pair. To normalize, in both features we use lemmas instead of surface forms. We discard features that occur in fewer than three mention pairs.

### 3.3 Crowd-Sourced Data

Crowd sourcing provides a cheap source of human labeling to improve the quality of our classifier. In this work, we specifically examine feedback on the result of distant supervision. Precisely, we construct the union of  $R_1^+ \cup \dots \cup R_N^+$  from Section 3.1. We then solicit human labeling from Mechanical Turk (MTurk) while applying state-of-the-art quality control protocols following Gormley et al. (2010) and those in the MTurk manual.<sup>10</sup>

These quality-control protocols are critical to ensure high quality: spamming is common on MTurk and some turkers may not be as proficient or careful as expected. To combat this, we replicate each question three times and, following Gormley

<sup>8</sup>We did not run Ensemble on ClueWeb because we had very few machines satisfying Ensemble’s memory requirement. In contrast, MaltParser requires less memory and we could leverage Condor (Thain et al., 2005) to parse ClueWeb with MaltParser within several days (using about 50K CPU hours).

<sup>9</sup>We experiment with a slightly more sophisticated entity-linking system as well, which resulted in higher overall quality. The results below are from the simple entity-linking system.

<sup>10</sup>[http://mturkpublic.s3.amazonaws.com/docs/MTURK\\_BP.pdf](http://mturkpublic.s3.amazonaws.com/docs/MTURK_BP.pdf)

<sup>7</sup><http://nlp.cs.qc.cuny.edu/kbp/2010/>



et al. (2010), plant gold-standard questions: each task consists of five yes/no questions, one of which comes from our gold-standard pool.<sup>11</sup> By retaining only those answers that are consistent with this protocol, we are able to filter responses that were not answered with care or competency. We only use answers from workers who display overall high consistency with the gold standard (i.e., correctly answering at least 80% of the gold-standard questions).

### 3.4 Statistical Modeling Issues

Following Mintz et al. (2009), we use logistic regression classifiers to represent relation extractors. However, while Mintz et al. use a single multi-class classifier for all relations, Hoffman et al. (2011) and use an independent binary classifier for each individual relation; the intuition is that a pair of mentions (or entities) might participate in multiple target relations. We experimented with both protocols; since relation overlapping is rare for TAC-KBP and there was little difference in result quality, we focus on the binary-classification approach using training examples constructed as described in Section 3.1.

We compensate for the different sizes of distant and human labeled examples by training an objective function that allows to tune the weight of human versus distant labeling. We separately tune this parameter for each training set (with cross validation), but found that the result quality was robust with respect to a broad range of parameter values.<sup>12</sup>

## 4 Experiments

We describe our experiments to test the hypotheses that the following two factors improve distant-supervision quality: increasing the

- (1) corpus size, and
- (2) the amount of crowd-sourced feedback.

We confirm hypothesis (1), but, surprisingly, are unable to confirm (2). Specifically, when using logistic regression to train relation extractors, increasing corpus size improves, consistently and significantly, the precision and recall produced by distant supervision, regardless of human feedback levels. Using the

<sup>11</sup>We obtain the gold standard from a separate MTurk submission by taking examples that at least 10 out of 11 turkers answered yes, and then negate half of these examples by altering the relation names (e.g., *spouse* to *sibling*).

<sup>12</sup>More details in our technical report (Zhang et al., 2012).

methodology described in Section 3, human feedback has limited impact on the precision and recall produced from distant supervision by itself.

### 4.1 Evaluation Metrics

Just as direct training data are scarce, ground truth for relation extraction is scarce as well. As a result, prior work mainly considers two types of evaluation methods: (1) randomly sample a small portion of predictions (e.g., top-k) and manually evaluate precision/recall; and (2) use a held-out portion of seed facts (usually Freebase) as a kind of “distant” ground truth. We replace manual evaluation with a standardized relation-extraction benchmark: TAC-KBP 2010. TAC-KBP asks for extractions of 46 relations on a given set of 100 entities. Interestingly, the Freebase held-out metric (Mintz et al., 2009; Yao et al., 2010; Hoffmann et al., 2011) turns out to be heavily biased toward distantly labeled data (e.g., increasing human feedback *hurts* precision; see Section 4.6).

### 4.2 Experimental Setup

Our first group of experiments use the 1.8M-doc TAC-KBP corpus for training. We exclude from it the 33K documents that contain query entities in the TAC-KBP metrics. There are two key parameters: the corpus size (#docs)  $M$  and human feedback budget (#examples)  $N$ . We perform different levels of down-sampling on the training corpus. On TAC, we use subsets with  $M = 10^3, 10^4, 10^5$ , and  $10^6$  documents respectively. For each value of  $M$ , we perform 30 independent trials of uniform sampling, with each trial resulting in a training corpus  $D_i^M$ ,  $1 \leq i \leq 30$ . For each training corpus  $D_i^M$ , we perform distant supervision to train a set of logistic regression classifiers. From the full corpus, distant supervision creates around 72K training examples.

To evaluate the impact of human feedback, we randomly sample 20K examples from the input corpus (we remove any portion of the corpus that is used in an evaluation). Then, we ask three different crowd workers to label each example as either positive or negative using the procedure described in Section 3.3. We retain only credible answers using the gold-standard method (see Section 3.3), and use them as the pool of human feedback that we run experiments with. About 46% of our human labels are negative. Denote by  $N$  the number of examples that

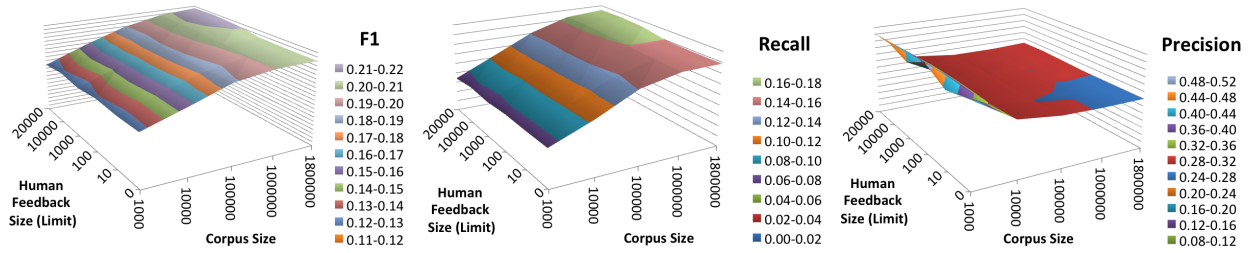


Figure 2: Impact of input sizes under the TAC-KBP metric, which uses documents mentioning 100 predefined entities as testing corpus with entity-level ground truth. We vary the sizes of the training corpus and human feedback while measuring the scores (F1, recall, and precision) on the TAC-KBP benchmark.

we want to incorporate human feedback for; we vary  $N$  in the range of 0, 10,  $10^2$ ,  $10^3$ ,  $10^4$ , and  $2 \times 10^4$ . For each selected corpus and value of  $N$ , we perform without-replacement sampling from examples of this corpus to select feedback for up to  $N$  examples. In our experiments, we found that on average an  $M$ -doc corpus contains about  $0.04M$  distant labels, out of which  $0.01M$  have human feedback. After incorporating human feedback, we evaluate the relation extractors on the TAC-KBP benchmark. We then compute the average F1, recall, and precision scores among all trials for each metric and each  $(M,N)$  pair. Besides the KBP metrics, we also evaluate each  $(M,N)$  pair using Freebase held-out data. Furthermore, we experiment with a much larger corpus: ClueWeb09. On ClueWeb09, we vary  $M$  over  $10^3, \dots, 10^8$ . Using the same metrics, we show at a larger scale that increasing corpus size can significantly improve both precision and recall.

### 4.3 Overall Impact of Input Sizes

We first present our experiment results on the TAC corpus. As shown in Figure 2, the F1 graph closely tracks the recall graph, which supports our earlier claim that quality is recall gated (Section 1). While increasing the corpus size improves F1 at a roughly log-linear rate, human feedback has little impact until both corpus size and human feedback size approach maximum  $M, N$  values. Table 1 shows the quality comparisons with minimum/maximum values of  $M$  and  $N$ .<sup>13</sup> We observe that increasing the corpus size significant improves per-relation recall

<sup>13</sup>When the corpus size is small, the total number of examples with feedback can be smaller than the budget size  $N$  – for example, when  $M = 10^3$  there are on average 10 examples with feedback even if  $N = 10^4$ .

	$M = 10^3$	$M = 1.8 \times 10^6$
$N = 0$	0.124	0.201
$N = 2 \times 10^4$	0.118	0.214

Table 1: TAC F1 scores with max/min values of  $M/N$ .

and F1 on 17 out of TAC-KBP’s 20 relations; in contrast, human feedback has little impact on recall, and only significantly improves the precision and F1 of 9 relations – while hurting F1 of 2 relations (i.e., `MemberOf` and `LivesInCountry`).<sup>14</sup>

(a) Impact of corpus size changes.

$M \setminus N$	0	10	$10^2$	$10^3$	$10^4$	$2e4$
$10^3 \rightarrow 10^4$	+	+	+	+	+	+
$10^4 \rightarrow 10^5$	+	+	+	+	+	+
$10^5 \rightarrow 10^6$	+	+	+	+	+	+
$10^6 \rightarrow 1.8e6$	0	0	0	+	+	+

(b) Impact of feedback size changes.

$N \setminus M$	$10^3$	$10^4$	$10^5$	$10^6$	$1.8e6$
$0 \rightarrow 10$	0	0	0	0	0
$10 \rightarrow 10^2$	0	0	0	+	+
$10^2 \rightarrow 10^3$	0	0	0	+	+
$10^3 \rightarrow 10^4$	0	0	0	0	+
$10^4 \rightarrow 2e4$	0	0	0	0	-
$0 \rightarrow 2e4$	0	0	0	+	+

Table 2: Two-tail t-test with d.f.=29 and  $p=0.05$  on the impact of corpus size and feedback size changes respectively. (We also tried  $p=0.01$ , which resulted in change of only a single cell in the two tables.) In (a), each column corresponds to a fixed human-feedback budget size  $N$ . Each row corresponds to a jump from one corpus size  $(M)$  to the immediate larger size. Each cell value indicates whether the TAC F1 metric changed significantly: + (resp. -) indicates that the quality increased (resp. decreased) significantly; 0 indicates that the quality did not change significantly. Table (b) is similar.

<sup>14</sup>We report more details on per-relation quality in our technical report (Zhang et al., 2012).

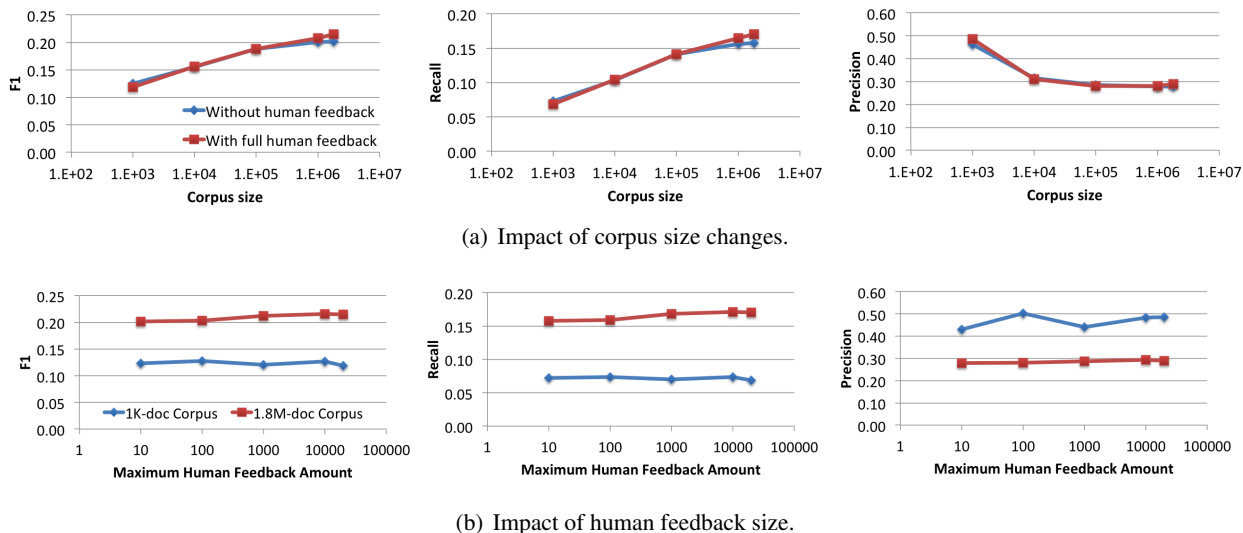


Figure 3: Projections of Figure 2 to show the impact of corpus size and human feedback amount on TAC-KBP F1, recall, and precision.

#### 4.4 Impact of Corpus Size

In Figure 3(a) we plot a projection of the graphs in Figure 2 to show the impact of corpus size on distant-supervision quality. The two curves correspond to when there is no human feedback and when we use all applicable human feedback. The fact that the two curves almost overlap indicates that human feedback had little impact on precision or recall. On the other hand, the quality improvement rate is roughly log-linear against the corpus size. Recall that each data point in Figure 2 is the average from 30 trials. To measure the statistical significance of changes in F1, we calculate t-test results to compare adjacent corpus size levels given each fixed human feedback level. As shown in Table 2(a), increasing the corpus size by a factor of 10 consistently and significantly improves F1. Although precision decreases as we use larger corpora, the decreasing trend is sub-log-linear and stops at around 100K docs. On the other hand, recall and F1 keep increasing at a log-linear rate.

#### 4.5 Impact of Human Feedback

Figure 3(b) provides another perspective on the results under the TAC metric: We fix a corpus size and plot the F1, recall, and precision as functions of human-feedback amount. Confirming the trend in Figure 2, we see that human feedback has little

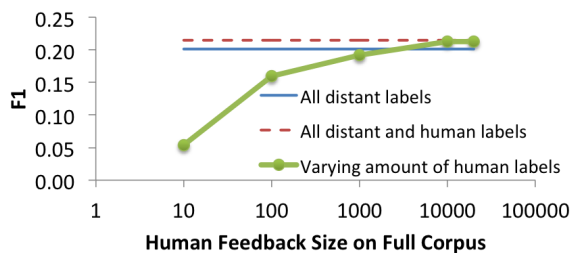


Figure 4: TAC-KBP quality of relation extractors trained using different amounts of human labels. The horizontal lines are comparison points.

impact on precision or recall with both corpus sizes.

We calculate t-tests to compare adjacent human feedback levels given each fixed corpus size level. Table 2(b)'s last row reports the comparison, for various corpus sizes (and, hence, number of distant labels), of (i) using no human feedback and (ii) using *all* of the human feedback we collected. When the corpus size is small (fewer than  $10^5$  docs), human feedback has no statistically significant impact on F1. The locations of '+'s suggest that the influence of human feedback becomes notable only when the corpus is very large (say with  $10^6$  docs). However, comparing the slopes of the curves in Figure 3(b) against Figure 3(a), the impact of human feedback is substantially smaller. The precision graph in Figure 3(b) suggests that human feedback does not no-

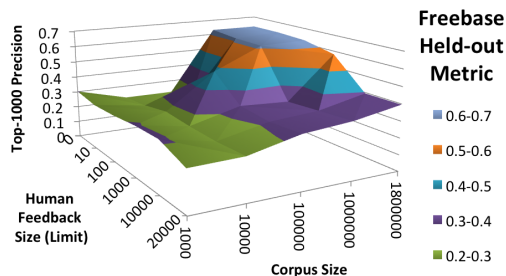


Figure 5: Impact of input sizes under the Freebase held-out metric. Note that the human feedback axis is in the reverse order compared to Figure 2.

tably improve precision on either the full corpus or on a small 1K-doc corpus. To assess the quality of human labels, we train extraction models with human labels only (on examples obtained from distant supervision). We vary the amount of human labels and plot the F1 changes in Figure 4. Although the F1 improves as we use more human labels, the best model has roughly the same performance as those trained from distant labels (with or without human labels). This suggests that the accuracy of human labels is not substantially better than distant labels.

#### 4.6 Freebase Held-out Metric

In addition to the TAC-KBP benchmark, we also follow prior work (Mintz et al., 2009; Yao et al., 2010; Hoffmann et al., 2011) and measure the quality using held-out data from Freebase. We randomly partition both Freebase and the corpus into two halves. One database-corpus pair is used for training and the other pair for testing. We evaluate the precision over the  $10^3$  highest-probability predictions on the test set. In Figure 5, we vary the size of the corpus in the train pair and the number of human labels; the precision reaches a dramatic peak when the corpus size is above  $10^5$  and uses little human feedback. This suggests that this Freebase held-out metric is biased toward solely relying on distant labels alone.

#### 4.7 Web-scale Corpora

To study how a Web corpus impacts distant-supervision quality, we select the first 100M English webpages from the ClueWeb09 dataset and measure how distant-supervision quality changes as we vary the number of webpages used. As shown in Figure 6, increasing the corpus size improves F1 up to

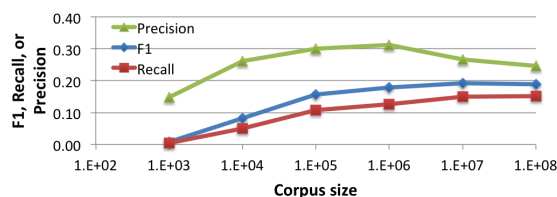


Figure 6: Impact of corpus size on the TAC-KBP quality with the ClueWeb dataset.

$10^7$  docs ( $p = 0.05$ ), while at  $10^8$  the two-tailed significance test reports no significant impact on F1 ( $p = 0.05$ ). The dip in precision in Figure 6 from  $10^6$  to either  $10^7$  or  $10^8$  is significant ( $p = 0.05$ ), and it is interesting future work to perform a detailed error analysis. Recall from Section 3 that to preprocess ClueWeb we use MaltParser instead of Ensemble. Thus, the F1 scores in Figure 6 are not comparable to those from the TAC training corpus.

## 5 Discussion and Conclusion

We study how the size of two types of cheaply available resources impact the precision and recall of distant supervision: (1) an unlabeled text corpus from which distantly labeled training examples can be extracted, and (2) crowd-sourced labels on training examples. We found that text corpus size has a stronger impact on precision and recall than human feedback. We observed that distant-supervision systems are often *recall gated*; thus, to improve distant-supervision quality, one should first try to enlarge the input training corpus and then increase precision.

It was initially counter-intuitive to us that human labels did not have a large impact on precision. One reason is that human labels acquired from crowd-sourcing have comparable noise level as distant labels – as shown by Figure 4. Thus, techniques that improve the accuracy of crowd-sourced answers are an interesting direction for future work. We used a particular form of human input (yes/no votes on distant labels) and a particular statistical model to incorporate this information (logistic regression). It is interesting future work to study other types of human input (e.g., new examples or features) and more sophisticated techniques for incorporating human input, as well as machine learning methods that explicitly model feature interactions.

## Acknowledgements

We gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, AFRL, or the US government. We are thankful for the generous support from the Center for High Throughput Computing, the Open Science Grid, and Miron Livny's Condor research group at UW-Madison. We are also grateful to Dan Weld for his insightful comments on the manuscript.

## References

- S. Brin. 1999. Extracting patterns and relations from the world wide web. In *Proceedings of The World Wide Web and Databases*, pages 172–183.
- A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. Hruschka Jr, and T. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Conference on Artificial Intelligence*, pages 1306–1313.
- J. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 363–370.
- M. Gormley, A. Gerber, M. Harper, and M. Dredze. 2010. Non-expert correction of automatically generated relation annotations. In *Proceedings of the NAACL HLT Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 204–207.
- M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th Conference on Computational Linguistics-Volume 2*, pages 539–545.
- R. Hoffmann, S. Amershi, K. Patel, F. Wu, J. Fogarty, and D.S. Weld. 2009. Amplifying community content creation with mixed initiative information extraction. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1849–1858. ACM.
- R. Hoffmann, C. Zhang, and D. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 286–295.
- R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 541–550.
- H. Ji, R. Grishman, H.T. Dang, K. Griffitt, and J. Ellis. 2010. Overview of the TAC 2010 knowledge base population track. In *Text Analysis Conference*.
- D. Lin and P. Pantel. 2001. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(4):343–360.
- M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1003–1011.
- T.V.T. Nguyen and A. Moschitti. 2011a. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceeding of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 277–282.
- T.V.T. Nguyen and A. Moschitti. 2011b. Joint distant and direct supervision for relation extraction. In *Proceeding of the International Joint Conference on Natural Language Processing*, pages 732–740.
- J. Nivre, J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov, and E. Marsi. 2007. Malt-parser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(02):95–135.
- S. Riedel, L. Yao, and A. McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part III*, pages 148–163.
- B. Settles. 2010. Active learning literature survey. Technical report, Computer Sciences Department, University of Wisconsin-Madison, USA.
- V.S. Sheng, F. Provost, and P.G. Ipeirotis. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622.
- M. Surdeanu and C. Manning. 2010. Ensemble models for dependency parsing: Cheap and good? In *Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 649–652.
- M. Surdeanu, D. McClosky, J. Tibshirani, J. Bauer, A.X. Chang, V.I. Spitzkovsky, and C. Manning. 2010. A simple distant supervision approach for the TAC-KBP slot filling task. In *Proceedings of Text Analysis Conference 2010 Workshop*.

- D. Thain, T. Tannenbaum, and M. Livny. 2005. Distributed computing in practice: The Condor experience. *Concurrency and Computation: Practice and Experience*, 17(2-4):323–356.
- R. Tibshirani. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288.
- F. Wu and D. Weld. 2007. Autonomously semantifying wikipedia. In *ACM Conference on Information and Knowledge Management*, pages 41–50.
- L. Yao, S. Riedel, and A. McCallum. 2010. Collective cross-document relation extraction without labelled data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1013–1023.
- C. Zhang, F. Niu, C. Ré, and J. Shavlik. 2012. Big data versus the crowd: Looking for relationships in all the right places (extended version). Technical report, Computer Sciences Department, University of Wisconsin-Madison, USA.

# Automatic Event Extraction with Structured Preference Modeling

Wei Lu and Dan Roth

University of Illinois at Urbana-Champaign

{luwei,danr}@illinois.edu

## Abstract

This paper presents a novel sequence labeling model based on the latent-variable semi-Markov conditional random fields for jointly extracting argument roles of events from texts. The model takes in coarse mention and type information and predicts argument roles for a given event template.

This paper addresses the event extraction problem in a primarily unsupervised setting, where no labeled training instances are available. Our key contribution is a novel learning framework called *structured preference modeling* (PM), that allows arbitrary preference to be assigned to certain structures during the learning procedure. We establish and discuss connections between this framework and other existing works. We show empirically that the structured preferences are crucial to the success of our task. Our model, trained without annotated data and with a small number of structured preferences, yields performance competitive to some baseline supervised approaches.

## 1 Introduction

Automatic template-filling-based event extraction is an important and challenging task. Consider the following text span that describes an “Attack” event:

... North Korea’s military may have fired a laser at a U.S. helicopter in March, a U.S. official said Tuesday, as the communist state ditched its last legal obligation to keep itself free of nuclear weapons ...

A partial event template for the “Attack” event is shown on the left of Figure 1. Each row shows an

argument for the event, together with a set of its acceptable mention types, where the type specifies a high-level semantic class a mention belongs to.

The task is to automatically fill the template entries with texts extracted from the text span above. The correct filling of the template for this particular example is shown on the right of Figure 1.

Performing such a task without any knowledge about the semantics of the texts is hard. One typical assumption is that certain coarse mention-level information, such as mention boundaries and their semantic class (a.k.a. *types*), are available. E.g.:

... [North Korea’s military]<sub>ORG</sub> may have fired [a laser]<sub>WEA</sub> at [a U.S. helicopter]<sub>VEH</sub> in [March]<sub>TME</sub>, a U.S. official said Tuesday, as the communist state ditched its last legal obligation to keep itself free of nuclear weapons ...

Such mention type information as shown on the left of Figure 1 can be obtained from various sources such as dictionaries, gazetteers, rule-based systems (Strötgen and Gertz, 2010), statistically trained classifiers (Ratinov and Roth, 2009), or some web resources such as Wikipedia (Ratinov et al., 2011).

However, in practice, outputs from existing mention identification and typing systems can be far from ideal. Instead of obtaining the above ideal annotation, one might observe the following noisy and ambiguous annotation for the given event span:

... [[North Korea’s]<sub>GPE|LOC</sub> military]<sub>ORG</sub> may have fired a laser at [a [U.S.]<sub>GPE|LOC</sub> helicopter]<sub>VEH</sub> in [March]<sub>TME</sub>, [a [U.S.]<sub>GPE|LOC</sub> official]<sub>PER</sub> said [Tuesday]<sub>TME</sub>, as [the communist state]<sub>ORG|FAC|LOC</sub> ditched its last legal obligation to keep [itself]<sub>ORG</sub> free of [nuclear weapons]<sub>WEA</sub> ...

Our task is to design a model to effectively select mentions in an event span and assign them with corresponding argument information, given such coarse

Argument	Possible Types	Extracted Text
ATTACKER	GPE, ORG, PER	<i>N. Korea's military</i>
INSTRUMENT	VEH, WEA	<i>a laser</i>
PLACE	FAC, GPE, LOC	-
TARGET	FAC, GPE, LOC ORG, PER, VEH	<i>a U.S. helicopter</i>
TIME-WITHIN	TME	<i>March</i>

Figure 1: The partial event template for the Attack event (left), and the correct event template annotation for the example event span given in Sec 1 (right). We primarily follow the ACE standard in defining arguments and types.

and often noisy mention type annotations.

This work addresses this problem by making the following contributions:

- Naturally, we are interested in identifying the active mentions (the mentions that serve as arguments) and their correct boundaries from the data. This motivates us to build a novel latent-variable semi-Markov conditional random fields model (Sarawagi and Cohen, 2004) for such an event extraction task. The learned model takes in coarse information as produced by existing mention identification and typing modules, and jointly outputs selected mentions and their corresponding argument roles.
- We address the problem in a more realistic scenario where annotated training instances are not available. We propose a novel general learning framework called *structured preference modeling* (or *preference modeling*, PM), which encompasses both the fully supervised and the latent-variable conditional models as special cases. The framework allows arbitrary declarative structured preference knowledge to be introduced to guide the learning procedure in a primarily unsupervised setting.

We present our semi-Markov model and discuss our preference modeling framework in Section 2 and 3 respectively. We then discuss the model’s relation with existing constraint-driven learning frameworks in Section 4. Finally, we demonstrate through experiments that structured preference information is crucial to model and present empirical results on a standard dataset in Section 5.

## 2 The Model

It is not hard to observe from the example presented in the previous section that dependencies between

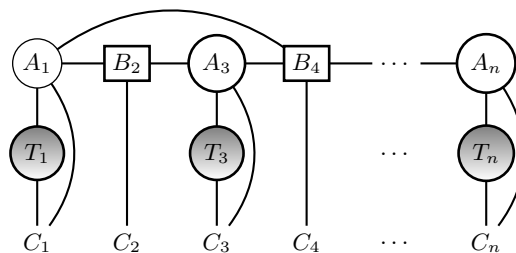


Figure 2: A simplified graphical illustration for the semi-Markov CRF, under a specific segmentation  $S \equiv \overline{C_1 C_2 \dots C_n}$ . In a supervised setting, only correct arguments are observed but their associated correct mention types are hidden (shaded).

arguments can be important and need to be properly modeled. This motivates us to build a joint model for extracting the event structures from the text.

We show a simplified graphical representation of our model in Figure 2. In the graph,  $C_1, C_2 \dots C_n$  refer to a particular segmentation of the event span, where  $C_1, C_3 \dots$  correspond to mentions (e.g., “North Korea’s military”, “a laser”) and  $C_2, C_4 \dots$  correspond to in-between mention word sequences (we call them *gaps*) (e.g., “may have fired”). The symbols  $T_1, T_3 \dots$  refer to mention types (e.g., GPE, ORG). The symbols  $A_1, A_3 \dots$  refer to event arguments that carry specific roles (e.g., ATTACKER). We also introduce symbols  $B_2, B_4 \dots$  to refer to inter-argument gaps. The event span is split into segments, where each segment is either linked to a mention type ( $T_i$ ; these segments can be referred to as “argument segments”), or directly linked to an inter-argument gap ( $B_j$ ; they can be referred to as “gap segments”). The two types of segments appear in the sequence in a strictly alternate manner, where the gaps can be of length zero. In the figure, for example, the segments  $C_1$  and  $C_3$  are identified as two argument segments (which are mentions of types  $T_1$  and  $T_3$  respectively) and are mapped to two “nodes”, and the segment  $C_2$  is identified as a gap segment that connects the two arguments  $A_1$  and  $A_3$ . Note that no overlapping arguments are allowed in this model<sup>1</sup>.

We use  $s$  to denote an event span and  $t$  to denote a specific realization (filling) of the event template. Templates consist of a set of arguments. Denote by  $h$  a particular mention boundary and type assignment for an event span, which gives us a specific segmentation of the given span. Following the conditional

<sup>1</sup>Extending the model to support certain argument overlapping is possible – we leave it for future work.



random fields model (Lafferty et al., 2001), we parameterize the conditional probability of the  $(t, h)$  pair given an event span  $s$  as follows:

$$P_{\Theta}(t, h|s) = \frac{e^{\mathbf{f}(s, h, t) \cdot \Theta}}{\sum_{t, h} e^{\mathbf{f}(s, h, t) \cdot \Theta}} \quad (1)$$

where  $\mathbf{f}$  gives the feature functions defined on the tuple  $(s, h, t)$ , and  $\Theta$  defines the parameter vector.

Our objective function is the logarithm of the joint conditional probability of observing the template realization for the observed event span  $s$ :

$$\begin{aligned} \mathcal{L}(\Theta) &= \sum_i \log P_{\Theta}(t_i|s_i) \\ &= \sum_i \log \frac{\sum_h e^{\mathbf{f}(s_i, h, t_i) \cdot \Theta}}{\sum_{t, h} e^{\mathbf{f}(s_i, h, t) \cdot \Theta}} \end{aligned} \quad (2)$$

This function is not convex due to the summation over the hidden variable  $h$ . To optimize it, we take its partial derivative with respect to  $\theta_j$ :

$$\begin{aligned} \frac{\partial \mathcal{L}(\Theta)}{\partial \theta_j} &= \sum_i \mathbb{E}_{p_{\Theta}(h|s_i, t_i)}[f_j(s_i, h, t_i)] \\ &\quad - \sum_i \mathbb{E}_{p_{\Theta}(t, h|s_i)}[f_j(s_i, h, t)] \end{aligned} \quad (3)$$

which requires computation of expectations terms under two different distributions. Such statistics can be collected efficiently with a forward-backward style algorithm in polynomial time (Okanohara et al., 2006). We will discuss the time complexity for our case in the next section.

Given its partial derivatives in Equation 3, one could optimize the objective function of Equation 2 with stochastic gradient ascent (LeCun et al., 1998) or L-BFGS (Liu and Nocedal, 1989). We choose to use L-BFGS for all our experiments in this paper.

Inference involves computing the most probable template realization  $t$  for a given event span:

$$\arg \max_t P_{\Theta}(t|s) = \arg \max_t \sum_h P_{\Theta}(t, h|s) \quad (4)$$

where the possible hidden assignments  $h$  need to be marginalized out. In this task, a particular realization  $t$  already uniquely defines a particular segmentation (mention boundaries) of the event span, thus the  $h$  only contributes type information to  $t$ . As we will discuss in Section 2.3, only a collection of local features are defined. Thus, a Viterbi-style dynamic programming algorithm is used to efficiently compute the desired solution.

## 2.1 Possible Segmentations

According to Equation 3, summing over all possible  $h$  is required. Since one primary assumption is that we have access to the output of existing mention identification and typing systems, the set of all possible mentions defines a lattice representation containing the set of all possible segmentations that comply with such mention-level information. Assuming there are  $A$  possible arguments for the event and  $K$  annotated mentions, the complexity of the forward-backward style algorithm is in  $O(A^3 K^2)$  under the “second-order” setting that we will discuss in Section 2.2. Typically,  $K$  is smaller than the number of words in the span, and the factor  $A^3$  can be regarded as a constant. Thus, the algorithm is very efficient.

As we have mentioned earlier, such coarse information, as produced by existing resources, could be highly ambiguous and noisy. Also, the output mentions can highly overlap with each other. For example, the phrase “North Korea” as in “North Korea’s military” can be assigned both type GPE and LOC, while “North Korea’s military” can be assigned the type ORG. Our model will need to disambiguate the mention boundaries as well as their types.

## 2.2 The Gap Segments

We believe the gap segments<sup>2</sup> are important to model since they can potentially capture dependencies between two or more adjacent arguments. For example, the word sequence “may have fired” clearly indicates an Attacker-Instrument relation between the two mentions “North Korea’s military” and “a laser”. Since we are only interested in modeling dependencies between adjacent argument segments, we assign hard labels to each gap segment based on its contextual argument information. Specifically, the label of each gap segment is uniquely determined by its surrounding argument segments with a list representation. For example, in a “first-order” setting, the gap segment that appears between its previous argument segment “ATTACKER” and its next argument segment “INSTRUMENT” is annotated as the list consisting of two elements: [ATTACKER, INSTRUMENT]. To capture longer-range dependencies, in this work we use a “second-order” setting (as shown in Figure 2),

<sup>2</sup>The length of a gap segment is arbitrary (including zero), unlike the seminal semi-Markov CRF model of Sarawagi and Cohen (2004).

which means each gap segment is annotated with a list that consists of its previous two argument segments as well as its subsequent one.

### 2.3 Features

Feature functions are factorized as products of two indicator functions: one defined on the input sequence (input features) and the other on the output labels (output features). In other words, we could re-write  $f_j(s, h, t)$  as  $f_k^{in}(s) \times f_l^{out}(h, t)$ .

For gap segments, we consider the following input feature templates:

- N-GRAM: Indicator function for  $n$ -gram appeared in the segment ( $n = 1, 2$ )
- ANCHOR: Indicator function for its relative position to the event anchor words (to the left, to the right, overlaps, contains)

and the following output feature templates:

- 1STORDER: Indicator function for the combination of its immediate left argument and its immediate right argument.
- 2NDOORDER: Indicator function for the combination of its immediate two left arguments and its immediate right argument.

For argument segments, we also define the same input feature templates as above, with the following additional ones to capture contextual information:

- CWORDS: Indicator function for the previous and next  $k$  ( $= 1, 2, 3$ ) words.
- CPOS: Indicator function for the previous and next  $k$  ( $= 1, 2, 3$ ) words' POS tags.

and we define the following output feature template:

- ARGTYPE: Indicator function for the combination of the argument and its associated type.

Although the semi-Markov CRF model gives us the flexibility in introducing features that can not be exploited in a standard CRF, such as entity name similarity scores and distance measures, in practice we found the above simple and general features work well. This way, the unnormalized score assigned to each structure is essentially a linear sum of the feature weights, each corresponding to an indicator function.

## 3 Learning without Annotated Data

The supervised model presented in the previous section requires substantial human efforts to annotate the training instances. Human annotations can be very expensive and sometimes impractical. Even if annotators are available, getting annotators to agree

with each other is often a difficult task in itself. Worse still, annotations often can not be reused: experimenting on a different domain or dataset typically require annotating new training instances for that particular domain or dataset.

We investigate inexpensive methods to alleviate this issue in this section. We introduce a novel general learning framework called *structured preference modeling*, which allows arbitrary prior knowledge about structures to be introduced to the learning process in a declarative manner.

### 3.1 Structured Preference Modeling

Denote by  $\mathcal{X}_\Omega$  and  $\mathcal{Y}_\Omega$  the entire input and output space, respectively. For a particular input  $x \in \mathcal{X}_\Omega$ , the set  $x \times \mathcal{Y}_\Omega$  gives us all possible structures that contain  $x$ . However, structures are not equally good. Some structures are generally regarded as better structures while some are worse.

Let's assume there is a function  $\kappa : \{x \times \mathcal{Y}_\Omega \rightarrow [0, 1]\}$  that measures the quality of the structures. This function returns the quality of a certain structure  $(x, y)$ , where the value 1 indicates a perfect structure, and 0 an impossible structure.

Under such an assumption, it is easy to observe that for a good structure  $(x, y)$ , we have  $p_\Theta(x, y) \times \kappa(x, y) = p_\Theta(x, y)$ , while for a bad structure  $(x, y)$ , we have  $p_\Theta(x, y) \times \kappa(x, y) = 0$ .

This motivates us to optimize the following objective function:

$$\mathcal{L}_u(\Theta) = \sum_i \log \frac{\sum_y p_\Theta(x_i, y) \times \kappa(x_i, y)}{\sum_y p_\Theta(x_i, y)} \quad (5)$$

Intuitively, optimizing such an objective function is equivalent to pushing the probability mass from bad structures to good structures corresponding to the same input.

When the preference function  $\kappa$  is defined as the indicator function for the correct structure  $(x_i, y_i)$ , the numerator terms of the above formula are simply of the forms  $p_\Theta(x_i, y_i)$ , and the model corresponds to the fully supervised CRF model.

The model also contains the latent-variable CRF as a special case. In a latent-variable CRF, we have input-output pairs  $(x_i, y_i)$ , but the underlying specific structure  $h$  that contains both  $x_i$  and  $y_i$  is hidden. The objective function is:

$$\sum_i \log \frac{\sum_h p_\Theta(x_i, h, y_i)}{\sum_{h, y'} p_\Theta(x_i, h, y')} \quad (6)$$

where  $p_{\Theta}(x_i, h, y_i) = 0$  unless  $h$  contains  $(x_i, y_i)$ . We define the following two functions:

$$q_{\Theta}(x_i, h) = \sum_{y'} p_{\Theta}(x_i, h, y') \quad (7)$$

$$\kappa(x_i, h) = \begin{cases} 1 & h \text{ contains } (x_i, y_i) \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

Note that this definition of  $\kappa$  models instance-specific preferences since it relies on  $y_i$ , which can be thought of as certain external prior knowledge related to  $x_i$ . It is easy to verify that  $p_{\Theta}(x_i, h, y_i) = q_{\Theta}(x_i, h) \times \kappa(x_i, h)$ , with  $q_{\Theta}$  remains a distribution. Thus, we could re-write the objective function as:

$$\sum_{i=1} \log \frac{\sum_h q_{\Theta}(x_i, h) \times \kappa(x_i, h)}{\sum_h q_{\Theta}(x_i, h)} \quad (9)$$

This shows that the latent-variable CRF is a special case of our objective function, with the above-defined  $\kappa$  function. Thus, this new objective function of Equation 5 is a generalization of both the supervised CRF and the latent-variable CRF.

The preference function  $\kappa$  serves as a source from which certain prior knowledge about the structure can be injected into our model in a principled way. Note that the function is defined at the complete structure level. This allows us to incorporate both local and arbitrary global structured information into the preference function.

Under the log-linear parameterization, we have:

$$\mathcal{L}'(\Theta) = \sum_i \log \frac{\sum_y e^{\mathbf{f}(x_i, y) \cdot \Theta} \times \kappa(x_i, y)}{\sum_y e^{\mathbf{f}(x_i, y) \cdot \Theta}} \quad (10)$$

This is again a non-convex optimization problem in general, and to solve it we take its partial derivative with respect to  $\theta_k$ :

$$\begin{aligned} \frac{\partial \mathcal{L}'(\Theta)}{\partial \theta_k} &= \sum_i \mathbb{E}_{p_{\Theta}(y|x_i; \kappa)} [f_k(x_i, y)] \\ &\quad - \sum_i \mathbb{E}_{p_{\Theta}(y|x_i)} [f_k(x_i, y)] \quad (11) \\ p_{\Theta}(y|x_i; \kappa) &\propto e^{\mathbf{f}(x_i, y) \cdot \Theta} \times \kappa(x_i, y) \\ p_{\Theta}(y|x_i) &\propto e^{\mathbf{f}(x_i, y) \cdot \Theta} \end{aligned}$$

### 3.2 Approximate Learning

Computation of the denominator terms of Equation 10 (and the second term of Equation 11) can be done

efficiently and exactly with dynamic programming. Our main concern is the computation of its numerator terms (and the first term of Equation 11).

The preference function  $\kappa$  is defined at the complete structure level. Unless the function is defined in specific forms that allow tractable dynamic programming (in the supervised case, which gives a unique term, or in the hidden variable case, which can define a packed representations of derivations), the efficient dynamic programming algorithm used by CRF is no longer generally applicable for arbitrary  $\kappa$ . In general, we resort to approximations.

In this work, we exploit a specific form of the preference function  $\kappa$ . We assume that there exists a projection from another decomposable function to  $\kappa$ . Specifically, we assume a collection of auxiliary functions, each of the form  $\kappa_p : (x, y) \rightarrow R$ , that scores a property  $p$  of the complete structure  $(x, y)$ . Each such function measures certain aspect of the quality of the structure. These functions assign positive scores to good structural properties and negative scores to bad ones. We then define  $\kappa(x, y) = 1$  for all structures that appear at the top- $n$  positions as ranked by  $\sum_p \kappa_p(x, y)$  for all possible  $y$ 's, and  $\kappa(x, y) = 0$  otherwise. We show some actual  $\kappa_p$  functions used for a particular event in Section 5.

At each iteration of the training process, to generate such a  $n$ -best list, we first use our model to produce top  $n \times b$  candidate outputs as scored by the current model parameters, and extract the top  $n$  outputs as scored by  $\sum_p \kappa_p(x, y)$ . In practice we set  $n = 10$  and  $b = 1000$ .

### 3.3 Event Extraction

Now we can obtain the objective function for our event extraction task. We replace  $x$  by  $s$  and  $y$  by  $(h, t)$  in Equation 10. This gives us the following function:

$$\mathcal{L}_u(\Theta) = \sum_i \log \frac{\sum_{t, h} e^{\mathbf{f}(s_i, h, t) \cdot \Theta} \times \kappa(s_i, h, t)}{\sum_{t, h} e^{\mathbf{f}(s_i, h, t) \cdot \Theta}} \quad (12)$$

The partial derivatives are as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}_u(\Theta)}{\partial \theta_k} &= \sum_i \mathbb{E}_{p_{\Theta}(t, h|s_i; \kappa)} [f_k(s_i, h, t)] \\ &\quad - \sum_i \mathbb{E}_{p_{\Theta}(t, h|s_i)} [f_k(s_i, h, t)] \quad (13) \\ p_{\Theta}(t, h|s_i; \kappa) &\propto e^{\mathbf{f}(s_i, h, t) \cdot \Theta} \times \kappa(s_i, h, t) \\ p_{\Theta}(t, h|s_i) &\propto e^{\mathbf{f}(s_i, h, t) \cdot \Theta} \end{aligned}$$

Recall that  $s$  is an event span,  $t$  is a specific realization of the event template, and  $h$  is the hidden mention information for the event span.

#### 4 Discussion: Preferences v.s. Constraints

Note that the objective function in Equation 5, if written in the additive form, leads to a cost function reminiscent of the one used in constraint-driven learning algorithm (CoDL) (Chang et al., 2007) (and similarly, posterior regularization (Ganchev et al., 2010), which we will discuss later at Section 6). Specifically, in CoDL, the following cost function is involved in its EM-like inference procedure:

$$\arg \max_y \Theta \cdot \mathbf{f}(x, y) - \rho \sum_c d(y, \mathcal{Y}_c) \quad (14)$$

where  $\mathcal{Y}_c$  defines the set of  $y$ 's that all satisfy a certain constraint  $c$ , and  $d$  defines a distance function from  $y$  to that set. The parameter  $\rho$  controls the degree of the penalty when constraints are violated.

There are some important distinctions between structured preference modeling (PM) and CoDL. CoDL primarily concerns *constraints*, which penalizes bad structures without explicitly rewarding good ones. On the other hand, PM concerns *preferences*, which can explicitly reward good structures.

Constraints are typically useful when one works on structured prediction problems for data with certain (often rigid) regularities, such as citations, advertisements, or POS tagging for complete sentences. In such tasks, desired structures typically present certain canonical forms. This allows declarative constraints to be specified as either local structure prototypes (e.g., in citation extraction, the word *pp.* always corresponds to the PAGES field, while *proceedings* is always associated with BOOKTITLE or JOURNAL), or as certain global regulations about complete structures (e.g., at least one word should be tagged as verb when performing a sentence-level POS tagging).

Unfortunately, imposing such (hard or soft) constraints for certain tasks such as ours, where the data tends to be of arbitrary forms without many rigid regularities, can be difficult and often inappropriate. For example, there is no guarantee that a certain argument will always be present in the event span, nor should a particular mention, if appeared, always be selected and assigned to a specific argument. For example, in the example event span given

in Section 1, both “*March*” and “*Tuesday*” are valid candidate mentions for the TIME-WITHIN argument given their annotated type TME. One important clue is that *March* appears after the word *in* and is located nearer to other mentions that can be potentially useful arguments. However, encoding such information as a general constraint can be inappropriate, as potentially better structures can be found if one considers other alternatives. On the other hand, if we believe the structural pattern “*at TARGET in TIME-WITHIN*” is in general considered a better sub-structure than “*said TIME-WITHIN*” for the “Attack” event, we may want to assign structured preference to a complete structure that contains the former, unless there exist other structured evidence showing the latter turns out to be better.

In this work, our preference function is related to another function that can be decomposed into a collection of property functions  $\kappa_p$ . Each of them scores a certain aspect of the complete structure. This formulation gives us a complete flexibility to assign arbitrary structured preferences, where positive scores can be assigned to good properties, and negative scores to bad ones. Thus, in this way, the quality of a complete structure is jointly measured with multiple different property functions.

To summarize, preferences are an effective way to “define” the event structure to the learner, which is essential in an unsupervised setting, which may not be easy to do with other forms of constraints. Preferences are naturally decomposable, which allows us to extend their impact without significantly effecting the complexity of inference.

## 5 Experiments

In this section, we present our experimental results on the standard ACE05<sup>3</sup> dataset (newswire portion). We choose to perform our evaluations on 4 events (namely, “Attack”, “Meet”, “Die” and “Transport”), which are the only events in this dataset that have more than 50 instances. For each event, we randomly split the instances into two portions, where 70% are used for learning, and the remaining 30% for evaluation. We list the corpus statistics in Table 2.

To present general results while making minimal assumptions, our primary event extraction results

<sup>3</sup><http://www.itl.nist.gov/iad/mig/tests/ace/2005/doc/>

Event	Without Annotated Training Data				With Annotated Training Data			
	Random	Unsup	Rule	PM	MaxEnt-b	MaxEnt-t	MaxEnt-p	semi-CRF
Attack	20.47	30.12	39.25	42.02	54.03	58.82	<b>65.18</b>	63.11
Meet	35.48	26.09	44.07	63.55	65.42	70.48	75.47	<b>76.64</b>
Die	30.03	13.04	40.58	55.38	51.61	59.65	63.18	<b>67.65</b>
Transport	20.40	6.11	44.34	57.29	53.76	57.63	61.02	<b>64.19</b>

Table 1: Performance for different events under different experimental settings, with gold mention boundaries and types. We report F1-measure percentages.

Event	#A	Learning Set		Evaluation Set		#P
		#I	#M	#I	#M	
Attack	8	188	300/509	78	121/228	<b>7</b>
Meet	7	57	134/244	24	52/98	<b>7</b>
Die	9	41	89/174	19	33/61	<b>6</b>
Transport	13	85	243/426	38	104/159	<b>6</b>

Table 2: Corpus statistics (#A: number of possible arguments for the event; #I: number of instances; #M: number of active/total mentions; #P: number of preference patterns used for performing our structured preference modeling.)

are independent of mention identification and typing modules, which are based on the gold mention information as given by the dataset. Additionally, we present results obtained by exploiting our in-house automatic mention identification and typing module, which is a hybrid system that combines statistical and rule-based approaches. The module’s statistical component is trained on the ACE04 dataset (newswire portion) and overall it achieves a micro-averaged F1-measure of 71.25% at our dataset.

### 5.1 With Annotated Training Data

With hand-annotated training data, we are able to train our model in a fully supervised manner. The right part of Table 1 shows the performance for the fully supervised models. For comparison, we present results from several alternative approaches based a collection of locally trained maximum entropy (MaxEnt) classifiers. In these approaches, we treat each argument of the template as one possible output class, plus a special “NONE” class for not selecting it as an argument. We train and apply the classifiers on argument segments (i.e., mentions) only. All the models are trained with the same feature set used in the semi-CRF model.

In the simplest baseline approach MaxEnt-b, type information for each mention is simply treated as one special feature. In the approach MaxEnt-t, we instead use the type information to constrain the

classifier’s predictions based on the acceptable types associated with each argument. This approach gives better performance than that of MaxEnt-b. This indicates that such locally trained classifiers are not robust enough to disambiguate arguments that take different types. As such, type information serving as additional constraints at the end does help.

To assess the importance of structured preference, we also perform experiments where structured preference information is incorporated at the inference time of the MaxEnt classifiers. Specifically, for each event, we first generate  $n$ -best lists for output structures. Next, we re-rank this list based on scores from our structured preference functions (we used the same preferences as to be discussed in the next section). The results for these approaches are given in the column of MaxEnt-p of Table 1. This simple approach gives us significant improvements, closing the gap between locally trained classifiers and the joint model (in one case the former even outperforms the latter). Note that no structured preference information is used when training and evaluating our semi-CRF model. This set of results is not surprising. In fact, similar observations are also reported in previous works when comparing joint model against local models with constraints incorporated (Roth and Yih, 2005). This clearly indicates that structured preference information is crucial to model.

### 5.2 Without Annotated Training Data

Now we turn to experiments for the more realistic scenario where human annotations are not available.

We first build our simplest baseline by randomly assigning arguments to each mention with mention type information serving as constraints. Averaged results over 1000 runs are reported in the first column of Table 1.

Since our model formulation leaves us with complete freedom in designing the preference function,

Type	Preference pattern ( $p$ )
General	$\{at in on\}$ followed by PLACE $\{during at in on\}$ followed by TIME-WITHIN
Die	AGENT (immediately) followed by $\{killed\}$ $\{killed\}$ (immediately) followed by VICTIM VICTIM (immediately) followed by $\{be\ killed\}$ AGENT followed by $\{killed\}$ (immediately) followed by VICTIM
Transport	X immediately followed by $\{, and\}$ immediately followed by X, where $X \in \{ORIGIN DESTINATION\}$ $\{from leave\}$ (immediately) followed by ORIGIN $\{at in to into\}$ immediately followed by DESTINATION PERSON followed by $\{to visit arrived\}$

Figure 3: The **complete list** of preference patterns used for the “Die” and “Transport” event. We simply set  $\kappa_p = 1.0$  for all  $p$ ’s. In other words, when a structure contains a pattern, its score is incremented by 1.0. We use  $\{ \}$  to refer to a set of possible words or arguments. For example,  $\{from|leave\}$  means a word which is either *from* or *leave*. The symbol  $()$  denotes optional. For example, “ $\{killed\}$  (immediately) followed by VICTIM” is equivalent to the following two preferences: “ $\{killed\}$  immediately followed by VICTIM”, and “ $\{killed\}$  followed by VICTIM”.

one could design arbitrarily good, domain-specific or even instance-specific preferences. However, to demonstrate its general effectiveness, in this work we only choose a minimal amount of general preference patterns for evaluations.

We make our preference patterns as general as possible. As shown in the last column (#P) of Table 2, we use only 7 preference patterns each for the “Attack” and “Meet” events, and 6 patterns each for the other two events. In Figure 3, we show the complete list of the 6 preference patterns for the “Die” and “Transport” event used for our experiments. Out of those 6 patterns, 2 are more general patterns shared across different events, and 4 are event-specific. In contrast, for example, for the “Die” event, the supervised approach requires human to select from 174 candidate mentions and annotate 89 of them.

Despite its simplicity, it works very well in practice. Results are given in the column of “PM” of Table 1. It generally gives competitive performance as compared to the supervised MaxEnt baselines.

On the other hand, a completely unsupervised approach where structured preferences are not specified, performs substantially worse. To run such completely unsupervised models, we essentially follow the same training procedure as that of the preference modeling, except that structured preference information is not in place when generating the  $n$ -best list. In the absence of proper guidances, such a procedure can easily converge to bad local minima. The results are reported in the “Unsup” column of Table 1. In practice, we found that very often, such a model would prefer short structures where many mentions are not selected as desired. As a result, the

unsupervised model without preference information can even perform worse than the random baseline<sup>4</sup>.

Finally, we also compare against an approach that regards the preferences as rules. All such rules are associated with a same weight and are used to jointly score each structure. We then output the structure that is assigned the highest total weight. Such an approach performs worse than our approach with preference modeling. The results are presented in the column of “Rule” of Table 1. This indicates that our model is able to learn to generalize with features through the guidance of our informative preferences. However, we also note that the performance of preference modeling depends on the actual quality and amount of preferences used for learning. In the extreme case, where only few preferences are used, the performance of preference modeling will be close to that of the unsupervised approach, while the rule-based approach will yield performance close to that of the random baseline.

The results with automatically predicted mention boundaries and types are given in Table 3. Similar observations can be made when comparing the performance of preference modeling with other approaches. This set of results further confirms the effectiveness of our approach using preference modeling for the event extraction task.

## 6 Related Work

Structured prediction with limited supervision is a popular topic in natural language processing.

<sup>4</sup>For each event, we only performed 1 run with all the initial feature weights set to zeros.

Event	Random	Unsup	PM	semi-CRF
Attack	14.26	26.19	32.89	<b>46.92</b>
Meet	26.65	14.08	45.28	<b>58.18</b>
Die	19.17	9.09	44.44	<b>48.57</b>
Transport	15.78	10.14	49.73	<b>52.34</b>

Table 3: Event extraction performance with automatic mention identifier and typer. We report F1 percentage scores for preference modeling (PM) as well as two baseline approaches. We also report performance of the supervised approach trained with the semi-CRF model for comparison.

Prototype driven learning (Haghighi and Klein, 2006) tackled the sequence labeling problem in a primarily unsupervised setting. In their work, a Markov random fields model was used, where some local constraints are specified via their *prototype list*.

Constraint-driven learning (CoDL) (Chang et al., 2007) and posterior regularization (PR) (Ganchev et al., 2010) are both primarily semi-supervised models. They define a constrained EM framework that regularizes posterior distribution at the E-step of each EM iteration, by pushing posterior distributions towards a constrained posterior set. We have already discussed CoDL in Section 4 and gave a comparison to our model. Unlike CoDL, in the PR framework constraints are relaxed to *expectation constraints*, in order to allow tractable dynamic programming. See also Samdani et al. (2012) for more discussions.

Contrastive estimation (CE) (Smith and Eisner, 2005a) is another log-linear framework for primarily unsupervised structured prediction. Their objective function is related to the pseudolikelihood estimator proposed by Besag (1975). One challenge is that it requires one to design a priori an effective neighborhood (which also needs to be designed in certain forms to allow efficient computation of the normalization terms) in order to obtain optimal performance. The model has been shown to work in unsupervised tasks such as POS induction (Smith and Eisner, 2005a), grammar induction (Smith and Eisner, 2005b), and morphological segmentation (Poon et al., 2009), where good neighborhoods can be identified. However, it is less intuitive what constitutes a good neighborhood in this task.

The neighborhood assumption of CE is relaxed in another latent structure approach (Chang et al., 2010a; Chang et al., 2010b) that focuses on semi-supervised learning with indirect supervisions, inspired by the CoDL model described above.

The locally normalized logistic regression (Berg-

Kirkpatrick et al., 2010) is another recently proposed framework for unsupervised structured prediction. Their model can be regarded as a generative model whose component multinomial is replaced with a miniature logistic regression where a rich set of local features can be incorporated. Empirically the model is effective in various unsupervised structured prediction tasks, and outperforms the globally normalized model. Although modeling the semi-Markov properties of our segments (especially the gap segments) in our task is potentially challenging, we plan to investigate in the future the feasibility for our task with such a framework.

## 7 Conclusions

In this paper, we present a novel model based on the semi-Markov conditional random fields for the challenging event extraction task. The model takes in coarse mention boundary and type information and predicts complete structures indicating the corresponding argument role for each mention.

To learn the model in an unsupervised manner, we further develop a novel learning approach called *structured preference modeling* that allows structured knowledge to be incorporated effectively in a declarative manner.

Empirically, we show that knowledge about structured preference is crucial to model and the preference modeling is an effective way to guide learning in this setting. Trained in a primarily unsupervised manner, our model incorporating structured preference information exhibits performance that is competitive to that of some supervised baseline approaches. Our event extraction system and code will be available for download from our group web page.

## Acknowledgments

We would like to thank Yee Seng Chan, Mark Sammons, and Quang Xuan Do for their help with the mention identification and typing system used in this paper. We gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, AFRL, or the US government.

## References

- T. Berg-Kirkpatrick, A. Bouchard-Côté, J. DeNero, and D. Klein. 2010. Painless unsupervised learning with features. In *Proc. of HLT-NAACL'10*, pages 582–590.
- J. Besag. 1975. Statistical analysis of non-lattice data. *The Statistician*, pages 179–195.
- M. Chang, L. Ratinov, and D. Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *Proc. of ACL'07*, pages 280–287.
- M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. 2010a. Discriminative learning over constrained latent representations. In *Proc. of NAACL'10*, 6.
- M. Chang, V. Srikumar, D. Goldwasser, and D. Roth. 2010b. Structured output learning with indirect supervision. In *Proc. ICML'10*.
- K. Ganchev, J. Graça, J. Gillenwater, and B. Taskar. 2010. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research (JMLR)*, 11:2001–2049.
- A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *Proc. of HLT-NAACL'06*, pages 320–327.
- J. D. Lafferty, A. McCallum, and F. C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML'01*, pages 282–289.
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, pages 2278–2324.
- D.C. Liu and J. Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528.
- D. Okanohara, Y. Miyao, Y. Tsuruoka, and J. Tsujii. 2006. Improving the scalability of semi-markov conditional random fields for named entity recognition. In *Proc. of ACL'06*, pages 465–472.
- H. Poon, C. Cherry, and K. Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proc. of HLT-NAACL'09*, pages 209–217.
- L. Ratinov and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proc. of CoNLL'09*, pages 147–155.
- L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proc. of ACL-HLT'11*, pages 1375–1384.
- D. Roth and W. Yih. 2005. Integer linear programming inference for conditional random fields. In *Proc. of ICML'05*, pages 736–743.
- R. Samdani, M. Chang, and D. Roth. 2012. Unified expectation maximization. In *Proc. NAACL'12*.
- S. Sarawagi and W.W. Cohen. 2004. Semi-markov conditional random fields for information extraction. *NIPS'04*, pages 1185–1192.
- N.A. Smith and J. Eisner. 2005a. Contrastive estimation: Training log-linear models on unlabeled data. In *Proc. of ACL'05*, pages 354–362.
- N.A. Smith and J. Eisner. 2005b. Guiding unsupervised grammar induction using contrastive estimation. In *Proc. of IJCAI Workshop on Grammatical Inference Applications*, pages 73–82.
- J. Strötgen and M. Gertz. 2010. Heideitime: High quality rule-based extraction and normalization of temporal expressions. In *Proc. of SemEval'10*, pages 321–324.



# Discriminative Learning for Joint Template Filling

**Einat Minkov**  
Information Systems  
University of Haifa  
Haifa 31905, Israel  
einatm@is.haifa.ac.il

**Luke Zettlemoyer**  
Computer Science & Engineering  
University of Washington  
Seattle, WA 98195, USA  
lsz@cs.washington.edu

## Abstract

This paper presents a joint model for template filling, where the goal is to automatically specify the fields of target relations such as seminar announcements or corporate acquisition events. The approach models mention detection, unification and field extraction in a flexible, feature-rich model that allows for joint modeling of interdependencies at all levels and across fields. Such an approach can, for example, learn likely event durations and the fact that start times should come before end times. While the joint inference space is large, we demonstrate effective learning with a Perceptron-style approach that uses simple, greedy beam decoding. Empirical results in two benchmark domains demonstrate consistently strong performance on both mention detection and template filling tasks.

## 1 Introduction

Information extraction (IE) systems recover structured information from text. Template filling is an IE task where the goal is to populate the fields of a target relation, for example to extract the attributes of a job posting (Califf and Mooney, 2003) or to recover the details of a corporate acquisition event from a news story (Freitag and McCallum, 2000).

This task is challenging due to the wide range of cues from the input documents, as well as non-textual background knowledge, that must be considered to find the best joint assignment for the fields of the extracted relation. For example, Figure 1 shows an extraction from CMU seminar announcement corpus (Freitag and McCallum, 2000). Here, the goal is to perform mention detection and extraction, by finding all of the text spans, or *mentions*,

```
<Koedinger@cmu.edu (Ken Koedinger).0.0.1.5.95.19.19.55>
Type: cmu.cs.scs
Topic: HCI seminar, Raj Reddy, 3:30 Friday 5-5, Wean 5409
Dates: 5-May-95
Time: 3:30
PostedBy: Koedinger on 1-May-95 at 19:19 from cmu.edu (Ken
Koedinger)
Abstract :

NOTE: DIFFERENT DAY AND TIME!!

Raj Reddy
3:30 Friday, May 5
Wean Hall 5409

"Some Necessary Conditions for a Good User Interface"

For our final Human-Computer Interaction seminar of the semester,
Raj Reddy will be presenting his thoughts on what makes a good
interface good. He hopes the discussion of "necessary conditions"
can serve as a source for new HCI research ideas. Should be a
thought-provoking way to transition to the summer!
```

Date	5/5/1995
Start Time	3:30PM
Location	Wean Hall 5409
Speaker	Raj Reddy
Title	Some Necessary Conditions for a Good User Interface
End Time	-

Figure 1: An example email and its template. Field mentions are highlighted in the text, grouped by color.

that describe field values, unify these mentions by grouping them according to target field, and normalizing the results within each group to provide the final extractions. Each of these steps requires significant knowledge about the target relation. For example, in Figure 1, the mention “3:30” appears three times and provides the only reference to a time. We must infer that this is the starting time, that the end time is never explicitly mentioned, and also that the event is in the afternoon. Such inferences may not hold in more general settings, such as extraction for medical emergencies or related events.

In this paper, we present a joint modeling and learning approach for the combined tasks of mention detection, unification, and template filling, as described above. As we will see in Section 2, previous work has mostly focused on learning tagging

models for mention detection, which can be difficult to aggregate into a full template extraction, or directly learning template field value extractors, often in isolation and with no reasoning across different fields in the same relation. We present a simple, feature-rich, discriminative model that readily incorporates a broad range of possible constraints on the mentions and joint field assignments.

Such an approach allows us to learn, for each target relation, an integrated model to weight the different extraction options, including for example the likely lengths for events, or the fact that start times should come before end times. However, there are significant computation challenges that come with this style of joint learning. We demonstrate empirically that these challenges can be solved with a combination of greedy beam decoding, performed directly in the joint space of possible mention clusters and field assignments, and structured Perceptron-style learning algorithm (Collins, 2002).

We report experimental evaluations on two benchmark datasets in different genres, the CMU seminar announcements and corporate acquisitions (Freitag and McCallum, 2000). In each case, we evaluated both template extraction and mention detection performance. Our joint learning approach provides consistently strong results across every setting, including new state-of-the-art results. We also demonstrate, through ablation studies on the feature set, the need for joint modeling and the relative importance of the different types of joint constraints.

## 2 Related Work

Research on the task of template filling has focused on the extraction of field value mentions from the underlying text. Typically, these values are extracted based on local evidence, where the most likely entity is assigned to each slot (Roth and Yih, 2001; Siefkes, 2008). There has been little effort towards a comprehensive approach that includes mention unification, as well as considers the structure of the target relational schema to create semantically valid outputs.

Recently, Haghighi and Klein (2010) presented a generative semi-supervised approach for template filling. In their model, slot-filling entities are first generated, and entity mentions are then realized in text. Thus, their approach performs coreference at

slot level. In addition to proper nouns (named entity mentions) that are considered in this work, they also account for nominal and pronominal noun mentions. This work presents a discriminative approach to this problem. An advantage of a discriminative framework is that it allows the incorporation of rich and possibly overlapping features. In addition, we enforce label consistency and semantic coherence at record level.

Other related works perform structured relation discovery for different settings of information extraction. In *open IE*, entities and relations may be inferred jointly (Roth and Yih, 2002; Yao et al., 2011). In this IE task, the target relation must agree with the entity types assigned to it; e.g., *born-in* relation requires a *place* as its argument. In addition, extracted relations may be required to be consistent with an existing ontology (Carlson et al., 2010). Compared with the extraction of tuples of entity mention pairs, template filling is associated with a more complex target relational schema.

Interestingly, several researchers have attempted to model label consistency and high-level relational constraints using state-of-the-art sequential models of named entity recognition (NER). Mainly, predetermined word-level dependencies were represented as links in the underlying graphical model (Sutton and McCallum, 2004; Finkel et al., 2005). Finkel *et al.* (2005) further modelled high-level semantic constraints; for example, using the CMU seminar announcements dataset, spans labeled as *start time* or *end time* were required to be semantically consistent. In the proposed framework we take a bottom-up approach to identifying entity mentions in text, where given a noisy set of candidate named entities, described using rich semantic and surface features, discriminative learning is applied to label these mentions. We will show that this approach yields better performance on the CMU seminar announcement dataset when evaluated in terms of NER. Our approach is complimentary to NER methods, as it can consolidate noisy overlapping predictions from multiple systems into coherent sets.

## 3 Problem Setting

In the template filling task, a target relation  $r$  is provided, comprised of attributes (also referred to as

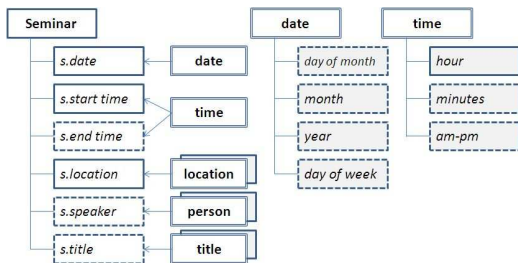


Figure 2: The relational schema for the seminars domain.

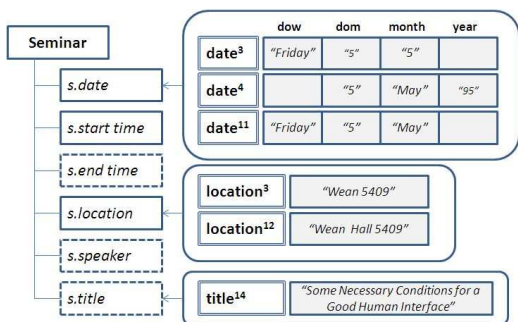


Figure 3: A record partially populated from text.

fields, or slots)  $A(r)$ . Given a document  $d$ , which is known to describe a tuple of the underlying relation, the goal is to populate the fields with values based on the text.

**The relational schema.** In this work, we describe domain knowledge through an extended relational database schema  $R$ . In this schema, every field of the target relation maps to a tuple of another relation, giving rise to a hierarchical view of template filling. Figure 2 describes a relational schema for the seminar announcement domain. As shown, each field of the *seminar* relation maps to another relation; e.g., *speaker*'s values correspond to *person* tuples. According to the outlined schema, most relations (e.g., *person*) consist of a single attribute, whereas the *date* and *time* relations are characterised with multiple attributes; for example, the *time* relation includes the fields of *hour*, *minutes* and *ampm*.

We will make use of limited domain knowledge, expressed as relation-level constraints that are typically realized in a database. Namely, the following tests are supported for each relation.

**Tuple validity.** This test reflects data integrity. The attributes of a relation may be defined as *mandatory* or *optional*. Mandatory attributes are denoted with a solid boundary in Figure 2 (e.g., *seminar.date*), and

optional attributes are denoted with a dashed boundary (e.g., *seminar.title*). Similar constraints can be posed on a set of attributes; e.g., either *day-of-month* or *day-of-week* must be populated in the *date* relation. Finally, a combination of field values may be required to be valid, e.g., the values of *day*, *month*, *year* and *day-of-week* must be consistent.

**Tuple contradiction.** This function checks whether two *valid* tuples  $v_1$  and  $v_2$  are inconsistent, implying a negation of possible unification of these tuples. In this work, we consider *date* and *time* tuples as contradictory if they contain semantically different values for some field; tuples of *location*, *person* and *title* are required to have minimal overlap in their string values to avoid contradiction.

**Template filling.** Given document  $d$ , the hierarchical schema  $R$  is populated in a bottom-up fashion. Generally, parent-free relations in the hierarchy correspond to generic entities, realized as entity mentions in the text. In Figure 2, these relations are denoted by double-line boundary, including *location*, *person*, *title*, *date* and *time*; every tuple of these relations maps to a named entity mention.<sup>1</sup>

Figure 3 demonstrates the correct mapping of named entity mentions to tuples, as well as tuple unification, for the example shown in Figure 1. For example, the mentions “Wean 5409” and “Wean Hall 5409” correspond to tuples of the *location* relation, where the two tuples are resolved into a unified set. To complete template filling, the remaining relations of the schema are populated bottom-up, where each field links to a unified set of populated tuples. For example, in Figure 3, the *seminar.location* field is linked to {“Wean Hall 5409”, “Wean 5409”}.

Value normalization of the unified tuples is another component of template filling. We partially address normalization: tuples of semantically detailed (multi-attribute) relations, e.g., *date* and *time*, are resolved into their semantic union, while textual tuples (e.g., *location*) are normalized to the longest string in the set. In this work, we assume that each template slot contains at most one value. This restriction can be removed, at the cost of increasing the size of the decoding search space.

<sup>1</sup>In the multi-attribute relations of *date* and *time*, each attribute maps to a text span, where the set of spans at tuple-level is required to be sequential (up to a small distance  $d$ ).

## 4 Structured Learning

Next, we describe how valid candidate extractions are instantiated (Sec. 4.1) and how learning is applied to assess the quality of the candidates (Sec. 4.2), where beam search is used to find the top scoring candidates efficiently (Sec. 4.3).

### 4.1 Candidate Generation

*Named entity recognition.* A set of candidate mentions  $S_d(a)$  is extracted from document  $d$  per each attribute  $a$  of a relation  $r \in L$ , where  $L$  is the set of parent-free relations in  $T$ . We aim at *high-recall* extractions; i.e.,  $S_d(a)$  is expected to contain the correct mentions with high probability. Various IE techniques, as well as an ensemble of methods, can be employed for this purpose. For each relation  $r \in L$ , *valid* candidate tuples  $E_d(r)$  are constructed from the candidate mentions that map to its attributes.

*Unification.* For every relation  $r \in L$ , we construct candidate sets of unified tuples,  $\{C_d(r) \subseteq E_d(r)\}$ . Naively, the number of candidate sets is exponential in the size of  $E_d(r)$ . Importantly, however, the tuples within a candidate unification set are required to be *non-contradictory*. In addition, the text spans that comprise the mentions within each set must not overlap. Finally, we do not split tuples with identical string values between different sets.

*Candidate tuples.* To construct the space of candidate tuples of the target relation, the remaining relations  $r \in \{T - L\}$  are visited bottom-up, where each field  $a \in A(r)$  is mapped in turn to a (possibly unified) populated tuple of its type. The valid (and non-overlapping) combinations of field mappings constitute a set of candidate tuples of  $r$ .

The candidate tuples generated using this procedure are structured entities, constructed using typed named entity recognition, unification, and hierarchical assignment of field values (Figure 3). We will derive features that describe local and global properties of the candidate tuples, encoding both surface and semantic information.

### 4.2 Learning

We employ a discriminative learning algorithm, following Collins (2002). Our goal is to find the candi-

### Algorithm 1: The beam search procedure

1. Populate every low-level relation  $r \in L$  from text  $d$ :
  - Construct a set of candidate valid tuples  $E_d(r)$  given high-recall typed candidate text spans  $S_d(a)$ ,  $a \in A(r)$ .
  - Group  $E_d(r)$  into possibly overlapping unified sets,  $\{C_d(r) \subseteq E_d(r)\}$ .
2. Iterate bottom-up through relations  $r \in \{T - L\}$ :
  - Initialize the set of candidate tuples  $E_d(r)$  to an empty set.
  - Iterate through attributes  $a \in A(r)$ :
    - Retrieve the set of candidate tuples (or unified tuple sets)  $E_d(r')$ , where  $r'$  is the relation that attribute  $a$  links to in  $T$ . Add an empty tuple to the set.
    - For every pair of candidate tuples  $e \in E_d(r)$  and  $e' \in E_d(r')$ , modify  $e$  by linking attribute  $a(e)$  to tuple  $e'$ .
    - Add the modified tuples, if valid, to  $E_d(r)$ .
    - Apply Equation 1 to rank the partially filled candidate tuples  $e \in E_d(r)$ . Keep the  $k$  top scoring candidates in  $E_d(r)$ , and discard the rest.
3. Apply Equation 1 to output a ranked list of extracted records  $E_d(r^*)$ , where  $r^*$  is the target relation.

date that maximizes:

$$F(y, \bar{\alpha}) = \sum_{j=1}^m \alpha_j f_j(y, d, T) \quad (1)$$

where  $f_j(d, y, T)$ ,  $j = 1, \dots, m$ , are pre-defined feature functions describing a candidate record  $y$  of the target relation given document  $d$  and the extended schema  $T$ . The parameter weights  $\alpha_j$  are to be learned from labeled instances. The training procedure involves initializing the weights  $\bar{\alpha}$  to zero. Given  $\bar{\alpha}$ , an inference procedure is applied to find the candidate that maximizes Equation 1. If the top-scoring candidate is different from the correct mapping known, then: (i)  $\bar{\alpha}$  is incremented with the feature vector of the correct candidate, and (ii) the feature vector of the top-scoring candidate is subtracted from  $\bar{\alpha}$ . This procedure is repeated for a fixed number of epochs. Following Collins, we employ the averaged Perceptron online algorithm (Collins, 2002; Freund and Schapire, 1999) for weight learning.

### 4.3 Beam Search

Unfortunately, optimal local decoding algorithms (such as the Viterbi algorithm in tagging problems (Collins, 2002)) can not be applied to our problem. We therefore propose using beam search to efficiently find the top scoring candidate. This means

that rather than instantiate the full space of valid candidate records (Section 4.1), we are interested in instantiating only those candidates that are likely to be assigned a high score by  $F$ . Algorithm 1 outlines the proposed beam search procedure. As detailed, only a set of top scoring tuples of size  $k$  (beam size) is maintained per relation  $r \in T$  during candidate generation. A given relation is populated incrementally, having each of its attributes  $a \in A(r)$  map in turn to populated tuples of its type, and using Equation 1 to find the  $k$  highest scoring *partially* populated tuples; this limits the number of candidate tuples evaluated to  $k^2$  per attribute, and to  $nk^2$  for a relation with  $n$  attributes. While beam search is efficient, performance may be compromised compared with an unconstrained search. The beam size  $k$  allows controlling the trade-off between performance and cost. An advantage of the proposed approach is that rather than output a single prediction, a list of coherent candidate tuples may be generated, ranked according to Equation 1.

## 5 Seminar Extraction Task

**Dataset** The CMU seminar announcement dataset (Freitag and McCallum, 2000) includes 485 emails containing seminar announcements. The dataset has been originally annotated with text spans referring to four slots: *speaker*, *location*, *stime*, and *etime*. We have annotated this dataset with two additional attributes: *date* and *title*.<sup>2</sup> We consider this corpus as an example of semi-structured text, where some of the field values appear in the email header, in a tabular structure, or using special formatting (Califf and Mooney, 1999; Minkov et al., 2005).<sup>3</sup>

We used a set of rules to extract candidate named entities per the types specified in Figure 2.<sup>4</sup> The rules encode information typically used in NER, including content and contextual patterns, as well as lookups in available dictionaries (Finkel et al., 2005; Minkov et al., 2005). The extracted candidates are high-recall and overlapping. In order to increase recall further, additional candidates were extracted based on document structure (Siefkes, 2008). The

<sup>2</sup>A modified dataset is available on the author’s homepage.

<sup>3</sup>Such structure varies across messages. Otherwise, the problem would reduce to wrapper learning (Zhu et al., 2006).

<sup>4</sup>The rule language used is based on cascaded finite state machines (Minorthird, 2008).

recall for the named entities of type *date* and *time* is near perfect, and is estimated at 96%, 91% and 90% for *location*, *speaker* and *title*, respectively.

**Features** The categories of the features used are described below. All features are binary and typed.<sup>5</sup>

*Lexical.* These features indicate the value and pattern of words within the text spans corresponding to each field. For example, lexical features per Figure 1 include *location.content.word.vean*, *location.pattern.capitalized*. Similar features are derived for a window of three words to the right and to the left of the included spans. In addition, we observe whether the words that comprise the text spans appear in relevant dictionaries: e.g., whether the spans assigned to the location field include words typical of location, such as “room” or “hall”. Lexical features of this form are commonly used in NER (Finkel et al., 2005; Minkov et al., 2005).

*Structural.* It has been previously shown that the structure available in semi-structured documents such as email messages is useful for information extraction (Minkov et al., 2005; Siefkes, 2008). As shown in Figure 1, an email message includes a header, specifying textual fields such as *topic*, *dates* and *time*. In addition, space lines and line breaks are used to emphasize blocks of important information. We propose a set of features that model correspondence between the text spans assigned to each field and document structure. Specifically, these features model whether at least one of the spans mapped to each field appears in the email header; captures a full line in the document; is indent; appears within space lines; or in a tabular format. In Figure 1, structural active features include *location.inHeader*, *location.fullLine*, *title.withinSpaceLines*, etc.

*Semantic.* These features refer to the semantic interpretation of field values. According to the relational schema (Figure 2), *date* and *time* include detailed attributes, whereas other relations are represented as strings. The semantic features encoded therefore refer to *date* and *time* only. Specifically, these features indicate whether a unified set of tuples defines a value for all attributes; for example, in Figure 1, the union of entities that map to the *date* field specify all of the attribute values of this relation, including *day-of-month*, *month*, *year*, and

<sup>5</sup>Real-value features were discretized into segments.

	Date	Stime	Etime	Location	Speaker	Title
Full model	96.1	99.3	98.7	96.4	87.5	69.5
No structural features	94.9	99.1	98.0	96.1	83.8	65.1
No semantic features	96.1	98.7	95.4	96.4	87.5	69.5
No unification	87.2	97.0	95.1	94.5	76.0	62.7
Individual fields	96.5	97.2	-	96.4	86.8	64.5

Table 1: Seminar extraction results (5-fold CV): Field-level F1

	Date	Stime	Etime	Location	Speaker	Title
SNOW (Roth and Yih, 2001)	-	<b>99.6</b>	96.3	75.2	73.8	-
BIEN (Peshkin and Pfeffer, 2003)	-	96.0	<b>98.8</b>	87.1	76.9	-
Elie (Finn, 2006)	-	98.5	96.4	86.5	<b>88.5</b>	-
TIE (Siefkes, 2008)	-	99.3	97.1	81.7	85.4	-
Full model	96.3	99.1	98.0	<b>96.9</b>	85.8	67.7

Table 2: Seminar extraction results (5-fold CV, trained on 50% of corpus): Field-level F1

*day-of-week*. Another feature encodes the size of the most semantically detailed named entity that maps to a field; for example, the most detailed entity mention of type *stime* in Figure 1 is “3:30”, comprising of two attribute values, namely *hour* and *minutes*. Similarly, the total number of semantic units included in a unified set is represented as a feature. These features were designed to favor semantically detailed mentions and unified sets. Finally, domain-specific semantic knowledge is encoded as features, including the *duration* of the seminar, and whether a *time* value is round (minutes divide by 5).

In addition to the features described, one may be interested in modeling cross-field information. We have experimented with features that encode the shortest distance between named entity mentions mapping to different fields (measured in terms of separating lines or sentences), based on the hypothesis that field values typically co-appear in the same segments of the document. These features were not included in the final model since their contribution was marginal. We leave further exploration of cross-field features in this domain to future work.

**Experiments** We conducted 5-fold cross validation experiments using the seminar extraction dataset. As discussed earlier, we assume that a single record is described in each document, and that each field corresponds to a single value. These assumptions are violated in a minority of cases. In evaluating the template filling task, only exact matches are accepted as true positives, where partial matches are counted as errors (Siefkes, 2008). Notably, the annotated labels as well as corpus itself are not error-free; for example, in some announcements the date and day-of-week specified are inconsistent.

Our evaluation is strict, where non-empty predicted values are counted as errors in such cases.

Table 1 shows the results of our full model using beam size  $k = 10$ , as well as model variants. In order to evaluate the contribution of the proposed features, we eliminated every feature group in turn. As shown in the table, removing the structural features hurt performance consistently across fields. In particular, structure is informative for the *title* field, which is otherwise characterised with low content and contextual regularity. Removal of the semantic features affected performance on the *stime* and *etime* fields, modeled by these features. In particular, the optional *etime* field, which has fewer occurrences in the dataset, benefits from modeling semantics.

An important question to be addressed in evaluation is to what extent the joint modeling approach contributes to performance. In another experiment we therefore mimic the typical scenario of template filling, in which the value of the highest scoring named entity is assigned to each field. In our framework, this corresponds to a setting in which a unified set includes no more than a single entity. The results are shown in Table 1 (‘no unification’). Due to reduced evidence given a single entity versus a coreferent set of entities, this results in significantly degraded performance. Finally, we experimented with populating every field of the target schema independently of the other fields. While results are overall comparable on most fields, this had negative impact on the *title* field. This is largely due to erroneous assignments of named entities of other types (mainly, *person*) as titles; such errors are avoided in the full joint model, where tuple validity is enforced.

Table 2 provides a comparison of the full model



	Date	Stime	Etime	Location	Speaker	Title
(Sutton and McCallum, 2004)	-	96.7	97.2	88.1	80.4	-
(Finkel et al., 2005)	-	<b>97.1</b>	<b>97.9</b>	90.0	84.2	-
Full model	95.4	<b>97.1</b>	<b>97.9</b>	<b>97.0</b>	<b>86.5</b>	75.5

Table 3: Seminar extraction results: Token-level F1

against previous state-of-the-art results. These results were all obtained using half of the corpus for training, and its remaining half for evaluation; the reported figures were averaged over five random splits. For comparison, we used 5-fold cross validation, where only a subset of each train fold that corresponds to 50% of the corpus was used for training. Due to the reduced training data, the results are slightly lower than in Table 1. (Note that we used the same test examples in both cases.) The best results per field are marked in boldface. The proposed approach yields the best or second-best performance on all target fields, and gives the best performance overall. While a variety of methods have been applied in previous works, none has modeled template filling in a joint fashion. As argued before, joint modeling is especially important for irregular fields, such as *title*; we provide first results on this field.

Previously, Sutton and McCallum (2004) and later Finkel *et-al.* (2005), applied sequential models to perform NER on this dataset, identifying named entities that pertain to the template slots. Both of these works incorporated coreference and high-level semantic information to a limited extent. We compare our approach to their work, having obtained and used the same 5-fold cross validation splits as both works. Table 3 shows results in terms of token F1. Our results evaluated on the named mention recognition task are superior overall, giving comparable or best performance on all fields. We believe that these results demonstrate the benefit of performing mention recognition as part of a joint model that takes into account detailed semantics of the underlying relational schema, when available.

Finally, we evaluate the *global* quality of the extracted records. Rather than assess performance at field-level, this stricter evaluation mode considers a whole tuple, requiring the values assigned to all of its fields to be correct. Overall, our full model (Table 1) extracts globally correct records for 52.6% of the examples. To our knowledge, this is the first work that provides this type of evaluation on this dataset. Importantly, an advantage of the proposed approach

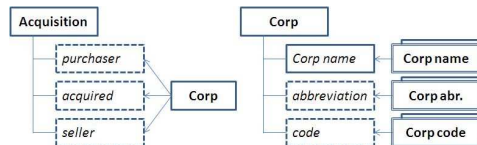


Figure 4: The relational schema for acquisitions.

is that it readily outputs a ranked list of coherent predictions. While the performance at the top of the output lists was roughly comparable, increasing  $k$  gives higher oracle recall: the correct record was included in the output  $k$ -top list 69.7%, 76.1% and 80.4% of the time, for  $k = 5, 10, 20$  respectively.

## 6 Corporate Acquisitions

**Dataset** The corporate acquisitions corpus contains 600 newswire articles, describing factual or potential corporate acquisition events. The corpus has been annotated with the official names of the parties to an acquisition: *acquired*, *purchaser* and *seller*, as well as their corresponding abbreviated names and company codes.<sup>6</sup> We describe the target schema using the relational structure depicted in Figure 4. The schema includes two relations: the *corp* relation describes a corporate entity, including its full name, abbreviated name and code as attributes; the target *acquisition* relation includes three role-designating attributes, each linked to a *corp* tuple.

Candidate name mentions in this strictly grammatical genre correspond to *noun phrases*. Documents were pre-processed to extract noun phrases, similarly to Haghighi and Klein (2010).

**Features** We model *syntactic* features, following Haghighi and Klein (2010). In order to compensate for parsing errors, shallow syntactic features were added, representing the values of neighboring verbs and prepositions (Cohen et al., 2005). While newswire documents are mostly unstructured, *structural* features are used to indicate whether any of the *purchaser*, *acquired* and *seller* text spans appears in

<sup>6</sup>In this work, we ignore other fields annotated, as they are inconsistently defined, have low number of occurrences in the corpus, and are loosely inter-related semantically.

	purname	purabr	purcode	acqname	acqabr	acqcode	sellname	sellabr	sellcode
TIE (batch)	<b>55.7</b>	<b>58.1</b>	-	<b>53.5</b>	55.0	-	31.8	25.8	-
TIE (inc)	51.6	55.3	-	49.2	51.7	-	26.0	24.0	-
Full model	48.9	55.0	<b>70.2</b>	50.7	<b>55.2</b>	<b>67.2</b>	<b>33.2</b>	<b>36.8</b>	<b>55.4</b>
<i>Model variants:</i>									
No inter-type and struct. ftrs	45.1	50.5	66.8	49.8	53.9	66.4	34.9	42.2	56.0
No semantic features	42.6	38.4	58.1	40.5	36.5	44.8	32.2	26.6	46.6
Individual roles	43.9	48.7	62.5	45.0	47.2	52.7	34.1	40.3	47.8

Table 4: Corp. acquisition extraction results: Field-level F1

	purname	purabr	purcode	acqname	acqabr	acqcode	sellname	sellabr	sellcode
TIE (batch)	<b>52.6</b>	40.5	-	<b>49.2</b>	43.7	-	28.7	16.4	-
TIE (inc)	48.4	38.6	-	44.7	42.7	-	23.6	14.5	-
Full model	45.0	<b>48.3</b>	<b>69.8</b>	46.4	<b>59.5</b>	<b>66.9</b>	<b>31.6</b>	<b>33.0</b>	<b>55.0</b>

Table 5: Corp. acquisition extraction results: Entity-level F1

the article’s header. *Semantic* features are applied to *corp* tuples: we model whether the abbreviated name is a subset of the full name; whether the corporate code forms exact initials of the full or abbreviated names; or whether it has high string similarity to any of these values. Finally, *cross-type features* encode the shortest string between spans mapping to different roles in the *acquisition* relation.

**Experiments** We applied beam search, where *corp* tuples are extracted first, and *acquisition* tuples are constructed using the top scoring *corp* entities. We used a default beam size  $k = 10$ . The dataset is split into a 300/300 train/test subsets.

Table 4 shows results of our full model in terms of field-level F1, compared against TIE, a state-of-the-art discriminative system (Siefkes, 2008). Unfortunately, we can not directly compare against a generative joint model evaluated on this dataset (Haghighi and Klein, 2010).<sup>7</sup> The best results per attribute are shown in boldface. Our full model performs better overall than TIE trained incrementally (similarly to our system), and is competitive with TIE using batch learning. Interestingly, the performance of our model on the *code* fields is high; these fields do not involve boundary prediction, and thus reflect the quality of role assignment.

Table 4 also shows the results of model variants. Removing the *inter type* and *structural* features mildly hurt performance, on average. In contrast, the *semantic* features, which account for the semantic cohesiveness of the populated *corp* tuples, are shown to be necessary. In particular, remov-

<sup>7</sup>They report average performance on a different set of fields; in addition, their results include modeling of pronouns and nominal mentions, which are not considered here.

ing them degrades the extraction of the abbreviated names; these features allow prediction of abbreviated names jointly with the full corporate names, which are more regular (e.g., include a distinctive suffix). Finally, we show results of predicting each role filler individually. Inferring the roles jointly (‘full model’) significantly improves performance.

Table 5 further shows results on NER, the task of recovering the sets of named entity mentions pertaining to each target field. As shown, the proposed joint approach performs overall significantly better than previous results reported. These results are consistent with the case study of seminar extraction.

## 7 Summary and Future Work

We presented a joint approach for template filling that models mention detection, unification, and field extraction in a flexible, feature-rich model. This approach allows for joint modeling of interdependencies at all levels and across fields. Despite the computational challenges of this joint inference space, we obtained effective learning with a Perceptron-style approach and simple beam decoding.

An interesting direction of future research is to apply reranking to the output list of candidate records using additional evidence, such as supporting evidence on the Web (Banko et al., 2008). Also, modeling additional features or feature combinations in this framework as well as effective feature selection or improved parameter estimation (Cramer et al., 2009) may boost performance. Finally, it is worth exploring scaling the approach to unrestricted event extraction, and jointly model extracting more than one relation per document.



## References

- Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2008. Open information extraction from the web. In *Proceedings of IJCAI*.
- Mary Elaine Califf and Raymond J. Mooney. 1999. Relational learning of pattern-match rules for information extraction. In *AAAI/IAAI*.
- Mary Elaine Califf and Raymond J. Mooney. 2003. Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, 4.
- Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of WSDM*.
- William W. Cohen, Einat Minkov, and Anthony Tomasic. 2005. Learning to understand web site update requests. In *Proceedings of the international joint conference on Artificial intelligence (IJCAI)*.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Koby Crammer, Alex Kulesza, and Mark Dredze. 2009. Adaptive regularization of weight vectors. In *Advances in Neural Information Processing Systems (NIPS)*.
- Jenny Rose Finkel, Trond Grenager, , and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of ACL*.
- Aidan Finn. 2006. A multi-level boundary classification approach to information extraction. In *PhD thesis*.
- Dayne Freitag and Andrew McCallum. 2000. Information extraction with hmm structures learned by stochastic optimization. In *AAAI/IAAI*.
- Yoav Freund and Rob Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3).
- Aria Haghighi and Dan Klein. 2010. An entity-level approach to information extraction. In *Proceedings of ACL*.
- Einat Minkov, Richard C. Wang, and William W. Cohen. 2005. Extracting personal names from emails: Applying named entity recognition to informal text. In *HLT/EMNLP*.
- Minorthird. 2008. Methods for identifying names and ontological relations in text using heuristics for inducing regularities from data. <http://http://minorthird.sourceforge.net>.
- Leonid Peshkin and Avi Pfeffer. 2003. Bayesian information extraction network. In *Proceedings of the international joint conference on Artificial intelligence (IJCAI)*.
- Dan Roth and Wen-tau Yih. 2001. Relational learning via propositional algorithms: An information extraction case study. In *Proceedings of the international joint conference on Artificial intelligence (IJCAI)*.
- Dan Roth and Wen-tau Yih. 2002. Probabilistic reasoning for entity and relation recognition. In *COLING*.
- Christian Siefkes. 2008. In *An Incrementally Trainable Statistical Approach to Information Extraction*. VDM Verlag.
- Charles Sutton and Andrew McCallum. 2004. Collective segmentation and labeling of distant entities in information extraction. In *Technical Report no. 04-49, University of Massachusetts*.
- Limin Yao, Aria Haghighi, Sebastian Riedel, and Andrew McCallum. 2011. Structured relation discovery using generative models. In *Proceedings of EMNLP*.
- Jun Zhu, Zaiqing Nie, Ji-Rong Wen, Bo Zhang, and Wei-Ying Ma. 2006. Simultaneous record detection and attribute labeling in web data extraction. In *Proc. of the ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD)*.

# Classifying French Verbs Using French and English Lexical Resources

**Ingrid Falk**  
Université de Lorraine/LORIA,  
Nancy, France  
ingrid.falk@loria.fr

**Claire Gardent**  
CNRS/LORIA,  
Nancy, France  
claire.gardent@loria.fr

**Jean-Charles Lamirel**  
Université de Strasbourg/LORIA,  
Nancy, France  
jean-charles.lamirel@loria.fr

## Abstract

We present a novel approach to the automatic acquisition of a Verbnets like classification of French verbs which involves the use (i) of a neural clustering method which associates clusters with features, (ii) of several supervised and unsupervised evaluation metrics and (iii) of various existing syntactic and semantic lexical resources. We evaluate our approach on an established test set and show that it outperforms previous related work with an F-measure of 0.70.

## 1 Introduction

Verb classifications have been shown to be useful both from a theoretical and from a practical perspective. From the theoretical viewpoint, they permit capturing syntactic and/or semantic generalisations about verbs (Levin, 1993; Kipper Schuler, 2006). From a practical perspective, they support factorisation and have been shown to be effective in various NLP (Natural language Processing) tasks such as semantic role labelling (Swier and Stevenson, 2005) or word sense disambiguation (Dang, 2004).

While there has been much work on automatically acquiring verb classes for English (Sun et al., 2010) and to a lesser extent for German (Brew and Schulte im Walde, 2002; Schulte im Walde, 2003; Schulte im Walde, 2006), Japanese (Oishi and Matsumoto, 1997) and Italian (Merlo et al., 2002), few studies have been conducted on the automatic classification of French verbs. Recently however, two proposals have been put forward.

On the one hand, (Sun et al., 2010) applied a clustering approach developed for English to French. They exploit features extracted from a large scale subcategorisation lexicon (LexSchem (Messiant, 2008)) acquired fully automatically from Le Monde newspaper corpus and show that, as for English, syntactic frames and verb selectional preferences perform better than lexical cooccurrence features. Their approach achieves a F-measure of 55.1 on 116 verbs occurring at least 150 times in LexSchem. The best performance is achieved when restricting the approach to verbs occurring at least 4000 times (43 verbs) with an F-measure of 65.4.

On the other hand, Falk and Gardent (2011) present a classification approach for French verbs based on the use of Formal Concept Analysis (FCA). FCA (Barbut and Monjardet, 1970) is a symbolic classification technique which permits creating classes associating sets of objects (eg. French verbs) with sets of features (eg. syntactic frames). Falk and Gardent (2011) provide no evaluation for their results however, only a qualitative analysis.

In this paper, we describe a novel approach to the clustering of French verbs which (i) gives good results on the established benchmark used in (Sun et al., 2010) and (ii) associates verbs with a feature profile describing their syntactic and semantic properties. The approach exploits a clustering method called IGNMF (Incremental Growing Neural Gas with Feature Maximisation, (Lamirel et al., 2011b)) which uses the features characterising each cluster both to guide the clustering process and to label the output clusters. We apply this method to the data contained in various verb lexicons and we evalu-

ate the resulting classification on a slightly modified version of the gold standard provided by (Sun et al., 2010). We show that the approach yields promising results (F-measure of 70%) and that the clustering produced systematically associates verbs with syntactic frames and thematic grids thereby providing an interesting basis for the creation and evaluation of a Verbnet-like classification.

Section 2 describes the lexical resources used for feature extraction and Section 3 the experimental setup. Sections 4 and 5 present the data used for and the results obtained. Section 6 concludes.

## 2 Lexical Resources Used

Our aim is to acquire a classification which covers the core verbs of French, could be used to support semantic role labelling and is similar in spirit to the English Verbnet. In this first experiment, we therefore favoured extracting the features used for clustering, not from a large corpus parsed automatically, but from manually validated resources<sup>1</sup>. These lexical resources are (i) a syntactic lexicon produced by merging three existing lexicons for French and (ii) the English Verbnet.

Among the many syntactic lexicons available for French (Nicolas et al., 2008; Messiant, 2008; Kupśc and Abeillé, 2008; van den Eynde and Mertens, 2003; Gross, 1975), we selected and merged three lexicons built or validated manually namely, Dicovalence, TreeLex and the LADL tables. The resulting lexicon contains 5918 verbs, 20433 lexical entries (i.e., verb/frame pairs) and 345 subcategorisation frames. It also contains more detailed syntactic and semantic features such as lexical preferences (e.g., locative argument, concrete object) or thematic role information (e.g., symmetric arguments, asset role) which we make use of for clustering.

We use the English Verbnet as a resource for associating French verbs with thematic grids as follows. We translate the verbs in the English Verbnet classes to French using English-French dictionaries<sup>2</sup>. To

<sup>1</sup>Of course, the same approach could be applied to corpus based data (as done e.g., in (Sun et al., 2010)) thus making the approach fully unsupervised and directly applicable to any language for which a parser is available.

<sup>2</sup>For the translation we use the following resources: Sci-Fran-Euradic, a French-English bilingual dictionary, built and improved by linguists (<http://catalog.elra.info/>

deal with polysemy, we train a supervised classifier as follows. We first map French verbs with English Verbnet classes: A French verb is associated with an English Verbnet class if, according to our dictionaries, it is a translation of an English verb in this class. The task of the classifier is then to produce a probability estimate for the correctness of this association, given the training data. The training set is built by stating for 1740 ⟨French verb, English Verbnet class⟩ pairs whether the verb has the thematic grid given by the pair’s Verbnet class<sup>3</sup>. This set is used to train an SVM (support vector machine) classifier<sup>4</sup>. The features we use are similar to those used in (Mouton, 2010): they are numeric and are derived for example from the number of translations an English or French verb had, the size of the Verbnet classes, the number of classes a verb is a member of etc. The resulting classifier gives for each ⟨French verb, English VN class⟩ pair the estimated probability of the pair’s verb being a member of the pair’s class<sup>5</sup>. We select 6000 pairs with highest probability estimates and obtain the translated classes by assigning each verb in a selected pair to the pair’s class. This way French verbs are effectively associated with one or more English Verbnet thematic grids.

## 3 Clustering Methods, Evaluation Metrics and Experimental Setup

### 3.1 Clustering Methods

The IGNGF clustering method is an incremental neural “winner-take-most” clustering method belonging to the family of the free topology neural clustering methods. Like other neural free topology methods such as Neural Gas (NG) (Martinetz and Schulen, 1991), Growing Neural Gas (GNG) (Fritzke, 1995), or Incremental Growing Neural Gas (IGNG) (Prudent and Ennaji, 2005), the IGNGF method makes use of Hebbian learning

[product\\_info.php?products\\_id=666](http://product_info.php?products_id=666)), Google dictionary (<http://www.google.com/dictionary>) and Dicovalence (van den Eynde and Mertens, 2003).

<sup>3</sup>The training data consists of the verbs and Verbnet classes used in the gold standard presented in (Sun et al., 2010).

<sup>4</sup>We used the libsvm (Chang and Lin, 2011) implementation of the classifier for this step.

<sup>5</sup>The accuracy of the classifier on the held out random test set of 100 pairs was of 90%.

(Hebb, 1949) for dynamically structuring the learning space. However, contrary to these methods, the use of a standard distance measure for determining a winner is replaced in IGNGF by feature maximisation. Feature maximisation is a cluster quality metric which associates each cluster with maximal features i.e., features whose Feature F-measure is maximal. Feature F-measure is the harmonic mean of Feature Recall and Feature Precision which in turn are defined as:

$$FR_c(f) = \frac{\sum_{v \in c} W_v^f}{\sum_{c' \in C} \sum_{v \in c'} W_v^f}, \quad FP_c(f) = \frac{\sum_{v \in c} W_v^f}{\sum_{f' \in F_c, v \in c} W_v^{f'}}$$

where  $W_x^f$  represents the weight of the feature  $f$  for element  $x$  and  $F_c$  designates the set of features associated with the verbs occurring in the cluster  $c$ . A feature is then said to be maximal for a given cluster iff its Feature F-measure is higher for that cluster than for any other cluster.

The IGNGF method was shown to outperform other usual neural and non neural methods for clustering tasks on relatively clean data (Lamirel et al., 2011b). Since we use features extracted from manually validated sources, this clustering technique seems a good fit for our application. In addition, the feature maximisation and cluster labeling performed by the IGNGF method has proved promising both for visualising clustering results (Lamirel et al., 2008) and for validating or optimising a clustering method (Attik et al., 2006). We make use of these processes in all our experiments and systematically compute cluster labelling and feature maximisation on the output clusterings. As we shall see, this permits distinguishing between clusterings with similar F-measure but lower “linguistic plausibility” (cf. Section 5). This facilitates clustering interpretation in that cluster labeling clearly indicates the association between clusters (verbs) and their prevalent features. And this supports the creation of a Verbnets style classification in that cluster labeling directly provides classes grouping together verbs, thematic grids and subcategorisation frames.

### 3.2 Evaluation metrics

We use several evaluation metrics which bear on different properties of the clustering.

**Modified Purity and Accuracy.** Following (Sun et al., 2010), we use modified purity (mPUR); weighted class accuracy (ACC) and F-measure to evaluate the clusterings produced. These are computed as follows. Each induced cluster is assigned the gold class (its *prevalent class*,  $\text{prev}(C)$ ) to which most of its member verbs belong. A verb is then said to be correct if the gold associates it with the prevalent class of the cluster it is in. Given this, purity is the ratio between the number of correct gold verbs in the clustering and the total number of gold verbs in the clustering<sup>6</sup>:

$$mPUR = \frac{\sum_{C \in \text{Clustering}, |\text{prev}(C)| > 1} |\text{prev}(C) \cap C|}{\text{Verbs}_{\text{Gold} \cap \text{Clustering}}},$$

where  $\text{Verbs}_{\text{Gold} \cap \text{Clustering}}$  is the total number of gold verbs in the clustering.

Accuracy represents the proportion of gold verbs in those clusters which are associated with a gold class, compared to all the gold verbs in the clustering. To compute accuracy we associate to each gold class  $C_{\text{Gold}}$  a dominant cluster, i.e. the cluster  $\text{dom}(C_{\text{Gold}})$  which has most verbs in common with the gold class. Then accuracy is given by the following formula:

$$ACC = \frac{\sum_{C \in \text{Gold}} |\text{dom}(C) \cap C|}{\text{Verbs}_{\text{Gold} \cap \text{Clustering}}}$$

Finally, F-measure is the harmonic mean of mPUR and ACC.

**Coverage.** To assess the extent to which a clustering matches the gold classification, we additionally compute the *coverage* of each clustering that is, the proportion of gold classes that are prevalent classes in the clustering.

**Cumulative Micro Precision (CMP).** As pointed out in (Lamirel et al., 2008; Attik et al., 2006), unsupervised evaluation metrics based on cluster labelling and feature maximisation can prove very useful for identifying the best clustering strategy. Following (Lamirel et al., 2011a), we use CMP to identify the best clustering. Computed on the clustering results, this metrics evaluates the quality of a clustering w.r.t. the cluster features rather than w.r.t.

<sup>6</sup>Clusters for which the prevalent class has only one element are ignored

to a gold standard. It was shown in (Ghribi et al., 2010) to be effective in detecting degenerated clustering results including a small number of large heterogeneous, “garbage” clusters and a big number of small size “chunk” clusters.

First, the *local Recall* ( $R_c^f$ ) and the *local Precision* ( $P_c^f$ ) of a feature  $f$  in a cluster  $c$  are defined as follows:

$$R_c^f = \frac{|v_c^f|}{|V^f|} \quad P_c^f = \frac{|v_c^f|}{|V_c|}$$

where  $v_c^f$  is the set of verbs having feature  $f$  in  $c$ ,  $V_c$  the set of verbs in  $c$  and  $V^f$ , the set of verbs with feature  $f$ .

Cumulative Micro-Precision (CMP) is then defined as follows:

$$CMP = \frac{\sum_{i=|C_{in f}|, |C_{sup}|} \frac{1}{|C_{i+}|^2} \sum_{c \in C_{i+}, f \in F_c} P_c^f}{\sum_{i=|C_{in f}|, |C_{sup}|} \frac{1}{C_{i+}}}$$

where  $C_{i+}$  represents the subset of clusters of  $C$  for which the number of associated verbs is greater than  $i$ , and:  $C_{in f} = \operatorname{argmin}_{c_i \in C} |c_i|$ ,  $C_{sup} = \operatorname{argmax}_{c_i \in C} |c_i|$

### 3.3 Cluster display, feature f-Measure and confidence score

To facilitate interpretation, clusters are displayed as illustrated in Table 1. Features are displayed in decreasing order of Feature F-measure (cf. Section 3.1) and features whose Feature F-measure is under the average Feature F-measure of the overall clustering are clearly delineated from others. In addition, for each verb in a cluster, a confidence score is displayed which is the ratio between the sum of the F-measures of its cluster maximised features over the sum of the F-measures of the overall cluster maximised features. Verbs whose confidence score is 0 are considered as orphan data.

### 3.4 Experimental setup

We applied an IDF-Norm weighting scheme (Robertson and Jones, 1976) to decrease the influence of the most frequent features (IDF component) and to compensate for discrepancies in feature number (normalisation).

C6- 14(14) [197(197)]

Prevalent Label — = AgExp-Cause

```
0.341100 G-AgExp-Cause
0.274864 C-SUJ:Ssub,OBJ:NP
0.061313 C-SUJ:Ssub
0.042544 C-SUJ:NP,DEOBJ:Ssub
*****
*****
0.017787 C-SUJ:NP,DEOBJ:VPinf
0.008108 C-SUJ:VPinf,AOBJ:PP
...
[**déprimer 0.934345 4(0)] [affliger 0.879122 3(0)]
[éblouir 0.879122 3(0)] [choquer 0.879122 3(0)]
[décevoir 0.879122 3(0)] [décontenancer 0.879122
3(0)] [décontracter 0.879122 3(0)] [désillusionner
0.879122 3(0)] [**ennuyer 0.879122 3(0)] [fasciner
0.879122 3(0)] [**heurter 0.879122 3(0)] ...
```

Table 1: Sample output for a cluster produced with the **grid-scf-sem** feature set and the IGNGF clustering method.

We use K-Means as a baseline. For each clustering method (K-Means and IGNGF), we let the number of clusters vary between 1 and 30 to obtain a partition that reaches an optimum F-measure and a number of clusters that is in the same order of magnitude as the initial number of Gold classes (i.e. 11 classes).

## 4 Features and Data

**Features** In the simplest case the features are the subcategorisation frames (**scf**) associated to the verbs by our lexicon. We also experiment with different combinations of additional, syntactic (**synt**) and semantic features (**sem**) extracted from the lexicon and with the thematic grids (**grid**) extracted from the English Verbnet.

The thematic grid information is derived from the English Verbnet as explained in Section 2. The syntactic features extracted from the lexicon are listed in Table 1(a). They indicate whether a verb accepts symmetric arguments (e.g., *John met Mary/John and Mary met*); has four or more arguments; combines with a predicative phrase (e.g., *John named Mary president*); takes a sentential complement or an optional object; or accepts the passive in *se* (similar to the English middle voice *Les habits se vendent bien/ The clothes sell well*). As shown in Table 1(a), these

(a) Additional syntactic features.

Feature	related VN class
Symmetric arguments	<i>amalgamate-22.2, correspond-36.1</i>
4 or more arguments	<i>get-13.5.1, send-11.1</i>
Predicate	<i>characterize-29.2</i>
Sentential argument	<i>correspond-36.1, characterize-29.2</i>
Optional object	implicit theme (Randall, 2010), p. 95
Passive built with <i>se</i>	theme role (Randall, 2010), p. 120

(b) Additional semantic features.

Feature	related VN class
Location role	<i>put-9.1, remove-10.1, ...</i>
Concrete object (non human role)	<i>hit-18.1</i> (eg. INSTRUMENT) <i>other.cos-45.4 ...</i>
Asset role	<i>get-13.5.1</i>
Plural role	<i>amalgamate-22.2, correspond-36.1</i>

Table 2: Additional syntactic (a) and semantic (b) features extracted from the LADL and Dicovalence resources and the alternations/roles they are possibly related to.

features are meant to help identify specific Verbnet classes and thematic roles. Finally, we extract four semantic features from the lexicon. These indicate whether a verb takes a locative or an asset argument and whether it requires a concrete object (non human role) or a plural role. The potential correlation between these features and Verbnet classes is given in Table 1(b).

**French Gold Standard** To evaluate our approach, we use the gold standard proposed by Sun et al. (2010). This resource consists of 16 fine grained Levin classes with 12 verbs each whose predominant sense in English belong to that class. Since our goal is to build a Verbnet like classification for French, we mapped the 16 Levin classes of the Sun et al. (2010)’s Gold Standard to 11 Verbnet classes thereby associating each class with a thematic grid. In addition we group Verbnet semantic roles as shown in Table 4. Table 3 shows the reference we use for evaluation.

**Verbs** For our clustering experiments we use the 2183 French verbs occurring in the translations of the 11 classes in the gold standard (cf. Section 4). Since we ignore verbs with only one feature the number of verbs and ⟨verb, feature⟩ pairs considered may vary slightly across experiments.

AgExp	Agent, Experiencer
AgentSym	Actor, Actor1, Actor2
Theme	Theme, Topic, Stimulus, Proposition
PredAtt	Predicate, Attribute
ThemeSym	Theme, Theme1, Theme2
Patient	Patient
PatientSym	Patient, Patient1, Patient2
Start	Material (transformation), Source (motion, transfer)
End	Product (transformation), Destination (motion), Recipient (transfer)
Location	
Instrument	
Cause	
Beneficiary	

Table 4: Verbnet role groups.

## 5 Results

### 5.1 Quantitative Analysis

Table 4(a) includes the evaluation results for all the feature sets when using IGNMF clustering.

In terms of F-measure, the results range from 0.61 to 0.70. This generally outperforms (Sun et al., 2010) whose best F-measures vary between 0.55 for verbs occurring at least 150 times in the training data and 0.65 for verbs occurring at least 4000 times in this training data. The results are not directly comparable however since the gold data is slightly different due to the grouping of Verbnet classes through their thematic grids.

In terms of features, the best results are obtained using the **grid-scf-sem** feature set with an F-measure of 0.70. Moreover, for this data set, the unsupervised evaluation metrics (cf. Section 3) highlight strong cluster cohesion with a number of clusters close to the number of gold classes (13 clusters for 11 gold classes); a low number of orphan verbs (i.e., verbs whose confidence score is zero); and a high Cumulated Micro Precision (CMP = 0.3) indicating homogeneous clusters in terms of maximising features. The coverage of 0.72 indicates that approximately 8 out of the 11 gold classes could be matched to a prevalent label. That is, 8 clusters were labelled with a prevalent label corresponding to 8 distinct gold classes.

In contrast, the classification obtained using the **scf-synt-sem** feature set has a higher CMP for the clustering with optimal mPUR (0.57); but a lower F-measure (0.61), a larger number of classes (16)

AgExp, PatientSym amalgamate-22.2: incorporer, associer, réunir, mélanger, mêler, unir, assembler, combiner, lier, fusionner
Cause, AgExp amuse-31.1: abattre, accabler, briser, déprimer, consterner, anéantir, épuiser, exténué, écraser, ennuyer, éreinter, inonder
AgExp, PredAtt, Theme characterize-29.2: appréhender, concevoir, considérer, décrire, définir, dépeindre, désigner, envisager, identifier, montrer, percevoir, représenter, ressentir
AgentSym, Theme correspond-36.1: coopérer, participer, collaborer, concourir, contribuer, associer
AgExp, Beneficiary, Extent, Start, Theme get-13.5.1: acheter, prendre, saisir, réserver, conserver, garder, préserver, maintenir, retenir, louer, affréter
AgExp, Instrument, Patient hit-18.1: cogner, heurter, battre, frapper, fouetter, taper, rosser, brutaliser, éreinter, maltraiter, corriger other_cos-45.4: mélanger, fusionner, consolider, renforcer, fortifier, adoucir, polir, atténuer, tempérer, pétrir, façonner, former
AgExp, Location, Theme light_emission-43.1 briller, étinceler, flamboyer, luire, resplendir, pétiller, rutiler, rayonner, scintiller modes_of_being_with_motion-47.3: trembler, frémir, osciller, vaciller, vibrer, tressaillir, frissonner, palpiter, grésiller, trembloter, palpiter run-51.3.2: voyager, aller, errer, circuler, courir, bouger, naviguer, passer, promener, déplacer
AgExp, End, Theme manner_speaking-37.3: râler, gronder, crier, ronchonner, grogner, bougonner, maugréer, rouspéter, grommeler, larmoyer, gémir, geindre, hurler, gueuler, brailler, chuchoter put-9.1: accrocher, déposer, mettre, placer, répartir, réintégrer, empiler, emporter, enfermer, insérer, installer say-37.7: dire, révéler, déclarer, signaler, indiquer, montrer, annoncer, répondre, affirmer, certifier, répliquer
AgExp, Theme peer-30.3: regarder, écouter, examiner, considérer, voir, scruter, dévisager
AgExp, Start, Theme remove-10.1: ôter, enlever, retirer, supprimer, retrancher, débarasser, soustraire, décompter, éliminer
AgExp, End, Start, Theme send-11.1: envoyer, lancer, transmettre, adresser, porter, expédier, transporter, jeter, renvoyer, livrer

Table 3: French gold classes and their member verbs presented in (Sun et al., 2010).

and a higher number of orphans (156). That is, this clustering has many clusters with strong feature cohesion but a class structure that markedly differs from the gold. Since there might be differences in structure between the English Verbnet and the thematic classification for French we are building, this is not necessarily incorrect however. Further investigation on a larger data set would be required to assess which clustering is in fact better given the data used and the classification searched for.

In general, data sets whose description includes semantic features (**sem** or **grid**) tend to produce better results than those that do not (**scf** or **synt**). This is in line with results from (Sun et al., 2010) which shows that semantic features help verb classification. It differs from it however in that the semantic features used by Sun et al. (2010) are selectional preferences while ours are thematic grids and a restricted set of manually encoded selectional preferences.

Noticeably, the **synt** feature degrades performance throughout: **grid,scf,synt** has lower F-measure than **grid,scf**; **scf,synt,sem** than **scf,sem**;

and **scf,synt** than **scf**. We have no clear explanation for this.

The best results are obtained with IGNGF method on most of the data sets. Table 4(b) illustrates the differences between the results obtained with IGNGF and those obtained with K-means on the **grid-sc-f-sem** data set (best data set). Although K-means and IGNGF optimal model reach similar F-measure and display a similar number of clusters, the very low CMP (0.10) of the K-means model shows that, despite a good Gold class coverage (0.81), K-means tend to produce more heterogeneous clusters in terms of features.

Table 4(b) also shows the impact of IDF feature weighting and feature vector normalisation on clustering. The benefit of preprocessing the data appears clearly. When neither IDF weighting nor vector normalisation are used, F-measure decreases from 0.70 to 0.68 and cumulative micro-precision from 0.30 to 0.21. When either normalisation or IDF weighting is left out, the cumulative micro-precision drops by up to 15 points (from 0.30 to 0.15 and 0.18) and the number of orphans increases from 67 up to 180.

(a) The impact of the feature set.

Feat. set	Nbr. feat.	Nbr. verbs	mPUR	ACC	F (Gold)	Nbr. classes	Cov.	Nbr. orphans	CMP at opt (13cl.)
scf	220	2085	0.93	0.48	0.64	17	0.55	129	0.28 (0.27)
grid, scf	231	2085	0.94	0.54	0.68	14	0.64	183	0.12 (0.12)
grid, scf, sem	237	2183	0.86	0.59	<b>0.70</b>	13	<b>0.72</b>	67	0.30 ( <b>0.30</b> )
grid, scf, synt	236	2150	0.87	0.50	0.63	14	0.72	66	0.13 (0.14)
grid, scf, synt, sem	242	2201	0.99	0.52	0.69	16	0.82	100	0.50 (0.22)
scf, sem	226	2183	0.83	0.55	0.66	23	0.64	146	0.40 (0.26)
scf, synt	225	2150	0.91	0.45	0.61	15	0.45	83	0.17 (0.22)
scf, synt, sem	231	2101	0.89	0.47	0.61	16	0.64	156	0.57 (0.11)

(b) Metrics for best performing clustering method (IGNGF) compared to K-means. Feature set is grid, scf, sem.

Method	mPUR	ACC	F (Gold)	Nbr. classes	Cov.	Nbr. orphans	CMP at opt (13cl.)
IGNGF with IDF and norm.	0.86	0.59	0.70	13	0.72	67	0.30 (0.30)
K-means with IDF and norm.	0.88	0.57	0.70	13	0.81	67	0.10 (0.10)
IGNGF, no IDF	0.86	0.59	0.70	17	0.81	126	0.18 (0.14)
IGNGF, no norm.	0.78	0.62	0.70	18	0.72	180	0.15 (0.11)
IGNGF, no IDF, no norm.	0.87	0.55	0.68	14	0.81	103	0.21 (0.21)

Table 5: Results. Cumulative micro precision (CMP) is given for the clustering at the mPUR optimum and in parentheses for 13 classes clustering.

That is, clusters are less coherent in terms of features.

## 5.2 Qualitative Analysis

We carried out a manual analysis of the clusters examining both the semantic coherence of each cluster (do the verbs in that cluster share a semantic component?) and the association between the thematic grids, the verbs and the syntactic frames provided by clustering.

**Semantic homogeneity:** To assess semantic homogeneity, we examined each cluster and sought to identify one or more Verbnet labels characterising the verbs contained in that cluster. From the 13 clusters produced by clustering, 11 clusters could be labelled. Table 6 shows these eleven clusters, the associated labels (abbreviated Verbnet class names), some example verbs, a sample sub-categorisation frame drawn from the cluster maximising features and an illustrating sentence. As can be seen, some clusters group together several subclasses and conversely, some Verbnet classes are spread over several clusters. This is not necessarily incorrect though. To start with, recall that we are aiming for a classification which groups together verbs with the same thematic grid. Given this, cluster C2 correctly groups together two Verbnet classes (other\_cos-45.4 and hit-18.1) which share the same thematic grid (cf. Table 3). In addition, the features

associated with this cluster indicate that verbs in these two classes are transitive, select a concrete object, and can be pronominalised which again is correct for most verbs in that cluster. Similarly, cluster C11 groups together verbs from two Verbnet classes with identical theta grid (light\_emission-43.1 and modes\_of\_being\_with\_motion-47.3) while its associated features correctly indicate that verbs from both classes accept both the intransitive form without object (*la jeune fille rayonne / the young girl glows, un cheval galope / a horse gallops*) and with a prepositional object (*la jeune fille rayonne de bonheur / the young girl glows with happiness, un cheval galope vers l’infini / a horse gallops to infinity*). The third cluster grouping together verbs from two Verbnet classes is C7 which contains mainly judgement verbs (to applaud, bless, compliment, punish) but also some verbs from the (very large) other\_cos-45.4 class. In this case, a prevalent shared feature is that both types of verbs accept a de-object that is, a prepositional object introduced by "de" (*Jean applaudit Marie d’avoir dansé / Jean applaudit Marie for having danced; Jean dégage le sable de la route / Jean clears the sand of the road*). The semantic features necessary to provide a finer grained analysis of their differences are lacking.

Interestingly, clustering also highlights classes which are semantically homogeneous but syntactically distinct. While clusters C6 and C10 both



contain mostly verbs from the amuse-31.1 class (*amuser, agacer, énerver, déprimer*), their features indicate that verbs in C10 accept the pronominal form (e.g., *Jean s’amuse*) while verbs in C6 do not (e.g., \**Jean se déprime*). In this case, clustering highlights a syntactic distinction which is present in French but not in English. In contrast, the dispersion of verbs from the other\_cos-45.4 class over clusters C2 and C7 has no obvious explanation. One reason might be that this class is rather large (361 verbs) and thus might contain French verbs that do not necessarily share properties with the original Verbnet class.

**Syntax and Semantics.** We examined whether the prevalent syntactic features labelling each cluster were compatible with the verbs and with the semantic class(es) manually assigned to the clusters. Table 6 sketches the relation between cluster, syntactic frames and Verbnet like classes. It shows for instance that the prevalent frame of the C0 class (manner\_speaking-37.3) correctly indicates that verbs in that cluster subcategorise for a sentential argument and an AOBJ (prepositional object in “à”) (e.g., *Jean bafouille à Marie qu’il est amoureux / Jean stammers to Mary that he is in love*); and that verbs in the C9 class (characterize-29.2) subcategorise for an object NP and an attribute (*Jean nomme Marie présidente / Jean appoints Marie president*). In general, we found that the prevalent frames associated with each cluster adequately characterise the syntax of that verb class.

## 6 Conclusion

We presented an approach to the automatic classification of french verbs which showed good results on an established testset and associates verb clusters with syntactic and semantic features.

Whether the features associated by the IINGF clustering with the verb clusters appropriately characterise these clusters remains an open question. We carried out a first evaluation using these features to label the syntactic arguments of verbs in a corpus with thematic roles and found that precision is high but recall low mainly because of polysemy: the frames and grids made available by the classification for a given verb are correct for that verb but not for the verb sense occurring in the corpus. This suggests that overlapping clustering techniques need to

C0	speaking: babiller, bafouiller, balbutier SUI:NP,OBJ:Ssub,AOBJ:PP <i>Jean bafouille à Marie qu’il l’aime / Jean stammers to Mary that he is in love</i>
C1	put: entasser, répandre, essaimer SUI:NP,POBJ:PP,DUMMY:REFL Loc, Plural <i>Les déchets s’entassent dans la cour / Waste piles in the yard</i>
C2	hit: broyer, démolir, fouetter SUI:NP,OBJ:NP T-Nhum <i>Ces pierres broient les graines / These stones grind the seeds.</i> other_cos: agrandir, alléger, amincir SUI:NP,DUMMY:REFL <i>les aéroports s’agrandissent sans arrêt / airports grow constantly</i>
C4	dedicate: s’engager à, s’obliger à, SUI:NP,AOBJ:VPinf,DUMMY:REFL <i>Cette promesse t’engage à nous suivre / This promise commits you to following us</i>
C5	conjecture: penser, attester, agréer SUI:NP,OBJ:Ssub <i>Le médecin atteste que l’employé n’est pas en état de travailler / The physician certifies that the employee is not able to work</i>
C6	amuse: déprimer, décontenancer, décevoir SUI:Ssub,OBJ:NP SUI:NP,DEOBJ:Ssub <i>Travailler déprime Marie / Working depresses Marie</i> <i>Marie déprime de ce que Jean parte / Marie depresses because of Jean’s leaving</i>
C7	other_cos: dégager, vider, drainer, sevrer judgement SUI:NP,OBJ:NP,DEOBJ:PP <i>vider le récipient de son contenu / empty the container of its contents</i> applaudir, bénir, blâmer, SUI:NP,OBJ:NP,DEOBJ:Ssub <i>Jean blame Marie d’avoir couru / Jean blames Mary for running</i>
C9	characterise: promouvoir, adouber, nommer SUI:NP,OBJ:NP,ATB:XP <i>Jean nomme Marie présidente / Jean appoints Marie president</i>
C10	amuse: agacer, amuser, enorgueillir SUI:NP,DEOBJ:XP,DUMMY:REFL <i>Jean s’enorgueillit d’être roi/ Jean is proud to be king</i>
C11	light: rayonner, clignoter, cliqueter SUI:NP,POBJ:PP <i>Jean clignote des yeux / Jean twinkles his eyes</i> motion: aller, passer, fuir, glisser SUI:NP,POBJ:PP <i>glisser sur le trottoir verglacé / slip on the icy sidewalk</i>
C12	transfer_msg: enseigner, permettre, interdire SUI:NP,OBJ:NP,AOBJ:PP <i>Jean enseigne l’anglais à Marie / Jean teaches Marie English.</i>

Table 6: Relations between clusters, syntactic frames and Verbnet like classes.

be applied.

We are also investigating how the approach scales up to the full set of verbs present in the lexicon. Both Dicovalence and the LADL tables contain rich detailed information about the syntactic and semantic properties of French verbs. We intend to tap on that potential and explore how well the various semantic features that can be extracted from these resources support automatic verb classification for the full set of verbs present in our lexicon.

## References

- M. Attik, S. Al Shehabi, and J.-C. Lamirel. 2006. Clustering Quality Measures for Data Samples with Multiple Labels. In *Databases and Applications*, pages 58–65.
- M. Barbut and B. Monjardet. 1970. *Ordre et Classification*. Hachette Université.
- C. Brew and S. Schulte im Walde. 2002. Spectral Clustering for German Verbs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 117–124, Philadelphia, PA.
- C. Chang and C. Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- H. T. Dang. 2004. *Investigations into the role of lexical semantics in word sense disambiguation*. Ph.D. thesis, U. Pennsylvania, US.
- I. Falk and C. Gardent. 2011. Combining Formal Concept Analysis and Translation to Assign Frames and Thematic Role Sets to French Verbs. In Amedeo Napoli and Vilem Vychodil, editors, *Concept Lattices and Their Applications*, Nancy, France, October.
- B. Fritzke. 1995. A growing neural gas network learns topologies. *Advances in Neural Information Processing Systems 7*, 7:625–632.
- M. Ghribi, P. Cuxac, J.-C. Lamirel, and A. Lelu. 2010. Mesures de qualité de clustering de documents : prise en compte de la distribution des mots clés. In Nicolas Béchet, editor, *Évaluation des méthodes d'Extraction de Connaissances dans les Données- EvalECD'2010*, pages 15–28, Hammamet, Tunisie, January. Fatiha Saïs.
- M. Gross. 1975. *Méthodes en syntaxe*. Hermann, Paris.
- D. O. Hebb. 1949. *The organization of behavior: a neuropsychological theory*. John Wiley & Sons, New York.
- K. Kipper Schuler. 2006. *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. Ph.D. thesis, University of Pennsylvania.
- A. Kupść and A. Abeillé. 2008. Growing treelex. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 4919 of *Lecture Notes in Computer Science*, pages 28–39. Springer Berlin / Heidelberg.
- J.-C. Lamirel, A. Phuong Ta, and M. Attik. 2008. Novel Labeling Strategies for Hierarchical Representation of Multidimensional Data Analysis Results. In *AIA - IASTED*, Innsbruck, Autriche.
- J. C. Lamirel, P. Cuxac, and R. Mall. 2011a. A new efficient and unbiased approach for clustering quality evaluation. In *QIMIE'11, PaKDD*, Shenzhen, China.
- J.-C. Lamirel, R. Mall, P. Cuxac, and G. Safi. 2011b. Variations to incremental growing neural gas algorithm based on label maximization. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 956–965.
- B. Levin. 1993. *English Verb Classes and Alternations: a preliminary investigation*. University of Chicago Press, Chicago and London.
- T. Martinetz and K. Schulten. 1991. A "Neural-Gas" Network Learns Topologies. *Artificial Neural Networks*, 1:397–402.
- P. Merlo, S. Stevenson, V. Tsang, and G. Allaria. 2002. A multilingual paradigm for automatic verb classification. In *ACL*, pages 207–214.
- C. Messiant. 2008. A subcategorization acquisition system for French verbs. In *Proceedings of the ACL-08: HLT Student Research Workshop*, pages 55–60, Columbus, Ohio, June. Association for Computational Linguistics.
- C. Mouton. 2010. *Ressources et méthodes semi-supervisées pour l'analyse sémantique de textes en français*. Ph.D. thesis, Université Paris 11 - Paris Sud UFR d'informatique.
- L. Nicolas, B. Sagot, É. de La Clergerie, and J. Farré. 2008. Computer aided correction and extension of a syntactic wide-coverage lexicon. In *Proc. of CoLing 2008*, Manchester, UK, August.
- A. Oishi and Y. Matsumoto. 1997. Detecting the organization of semantic subclasses of Japanese verbs. *International Journal of Corpus Linguistics*, 2(1):65–89, october.
- Y. Prudent and A. Ennaji. 2005. An incremental growing neural gas learns topologies. In *Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 1211–1216.
- J. H. Randall. 2010. *Linking*. Studies in Natural Language and Linguistic Theory. Springer, Dordrecht.
- S. E. Robertson and K. S. Jones. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146.
- S. Schulte im Walde. 2003. *Experiments on the Automatic Induction of German Semantic Verb Classes*. Ph.D. thesis, Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart. Published as AIMS Report 9(2).
- S. Schulte im Walde. 2006. Experiments on the automatic induction of german semantic verb classes. *Computational Linguistics*, 32(2):159–194.
- L. Sun, A. Korhonen, T. Poibeau, and C. Messiant. 2010. Investigating the cross-linguistic potential of verbnet: style classification. In *Proceedings of the 23rd International Conference on Computational Linguistics*,

- COLING '10, pages 1056–1064, Stroudsburg, PA, USA. Association for Computational Linguistics.
- R. S. Swier and S. Stevenson. 2005. Exploiting a verb lexicon in automatic semantic role labelling. In *HLT/EMNLP*. The Association for Computational Linguistics.
- K. van den Eynde and P. Mertens. 2003. La valence : l'approche pronominale et son application au lexique verbal. *Journal of French Language Studies*, 13:63–104.

# Modeling Sentences in the Latent Space

**Weiwei Guo**

Department of Computer Science,  
Columbia University,  
weiwei@cs.columbia.edu

**Mona Diab**

Center for Computational Learning Systems,  
Columbia University,  
mdiab@ccls.columbia.edu

## Abstract

Sentence Similarity is the process of computing a similarity score between two sentences. Previous sentence similarity work finds that latent semantics approaches to the problem do not perform well due to insufficient information in single sentences. In this paper, we show that by carefully handling words that are **not** in the sentences (missing words), we can train a reliable latent variable model on sentences. In the process, we propose a new evaluation framework for sentence similarity: Concept Definition Retrieval. The new framework allows for large scale tuning and testing of Sentence Similarity models. Experiments on the new task and previous data sets show significant improvement of our model over baselines and other traditional latent variable models. Our results indicate comparable and even better performance than current state of the art systems addressing the problem of sentence similarity.

## 1 Introduction

Identifying the degree of semantic similarity [SS] between two sentences is at the core of many NLP applications that focus on sentence level semantics such as Machine Translation (Kauchak and Barzilay, 2006), Summarization (Zhou et al., 2006), Text Coherence Detection (Lapata and Barzilay, 2005), etc. To date, almost all Sentence Similarity [SS] approaches work in the high-dimensional word space and rely mainly on word similarity. There are two main (not unrelated) disadvantages to word similarity based approaches: 1. lexical ambiguity as the pairwise word similarity ignores the semantic interaction between the word and its sentential context;

2. word co-occurrence information is not sufficiently exploited.

Latent variable models, such as Latent Semantic Analysis [LSA] (Landauer et al., 1998), Probabilistic Latent Semantic Analysis [PLSA] (Hofmann, 1999), Latent Dirichlet Allocation [LDA] (Blei et al., 2003) can solve the two issues naturally by modeling the semantics of words and sentences simultaneously in the low-dimensional latent space. However, attempts at addressing SS using LSA perform significantly below high dimensional word similarity based models (Mihalcea et al., 2006; O’Shea et al., 2008).

We believe that the latent semantics approaches applied to date to the SS problem have not yielded positive results due to the deficient modeling of the sparsity in the semantic space. SS operates in a very limited contextual setting where the sentences are typically very short to derive robust latent semantics. Apart from the SS setting, robust modeling of the latent semantics of short sentences/texts is becoming a pressing need due to the pervasive presence of more bursty data sets such as Twitter feeds and SMS where short contexts are an inherent characteristic of the data.

In this paper, we propose to model the missing words (words that are not in the sentence), a feature that is typically overlooked in the text modeling literature, to address the sparseness issue for the SS task. We define the missing words of a sentence as the whole vocabulary in a corpus minus the observed words in the sentence. Our intuition is since observed words in a sentence are too few to tell us what the sentence is about, missing words can be used to tell us what the sentence is **not** about. We assume that the semantic space of both the observed

and missing words make up the complete semantics profile of a sentence.

After analyzing the way traditional latent variable models (LSA, PLSA/LDA) handle missing words, we decide to model sentences using a weighted matrix factorization approach (Srebro and Jaakkola, 2003), which allows us to treat observed words and missing words differently. We handle missing words using a weighting scheme that distinguishes missing words from observed words yielding robust latent vectors for sentences.

Since we use a feature that is already implied by the text itself, our approach is very general (similar to LSA/LDA) in that it can be applied to any format of short texts. In contrast, existing work on modeling short texts focuses on exploiting additional data, e.g., Ramage et al. (2010) model tweets using their metadata (author, hashtag, etc.).

Moreover in this paper, we introduce a new evaluation framework for SS: Concept Definition Retrieval (CDR). Compared to existing data sets, the CDR data set allows for large scale tuning and testing of SS modules without further human annotation.

## 2 Limitations of Topic Models and LSA for Modeling Sentences

Usually latent variable models aim to find a latent semantic profile for a sentence that is most relevant to the observed words. By explicitly modeling missing words, we set another criterion to the latent semantics profile: it should **not** be related to the missing words in the sentence. However, missing words are not as informative as observed words, hence the need for a model that does a good job of modeling missing words at the right level of **emphasis/impact** is central to completing the semantic picture for a sentence.

LSA and PLSA/LDA work on a word-sentence co-occurrence matrix. Given a corpus, the row entries of the matrix are the unique  $M$  words in the corpus, and the  $N$  columns are the sentence ids. The yielded  $M \times N$  co-occurrence matrix  $X$  comprises the TF-IDF values in each  $X_{ij}$  cell, namely that TF-IDF value of word  $w_i$  in sentence  $s_j$ . For ease of exposition, we will illustrate the problem using a special case of the SS framework where the sentences are concept definitions in a dictionary such

as WordNet (Fellbaum, 1998) (WN). Therefore, the sentence corresponding to the concept definition of *bank#n#1* is a sparse vector in  $X$  containing the following observed words where  $X_{ij} \neq 0$ :

*the* 0.1, *financial* 5.5, *institution* 4, *that* 0.2, *accept* 2.1, *deposit* 3, and 0.1, *channel* 6, *the* 0.1, *money* 5, *into* 0.3, *lend* 3.5, *activity* 3

All the other words (*girl, car, ..., check, loan, business, ...*) in matrix  $X$  that do not occur in the concept definition are considered missing words for the concept entry *bank#n#1*, thereby their  $X_{ij} = 0$ .

Topic models (PLSA/LDA) do not explicitly model missing words. PLSA assumes each document has a distribution over  $K$  topics  $P(z_k|d_j)$ , and each topic has a distribution over all vocabularies  $P(w_i|z_k)$ . Therefore, PLSA finds a topic distribution for each concept definition that maximizes the log likelihood of the corpus  $X$  (LDA has a similar form):

$$\sum_i \sum_j X_{ij} \log \sum_k P(z_k|d_j) P(w_i|z_k) \quad (1)$$

In this formulation, missing words do not contribute to the estimation of sentence semantics, i.e., excluding missing words ( $X_{ij} = 0$ ) in equation 1 does not make a difference.

However, empirical results show that given a small number of observed words, usually topic models can only find one topic (most evident topic) for a sentence, e.g., the concept definitions of *bank#n#1* and *stock#n#1* are assigned the financial topic only without any further discernability. This results in many sentences are assigned exactly the same semantics profile as long as they are pertaining/mentioned within the same domain/topic. The reason is topic models try to learn a 100-dimension latent vector (assume dimension  $K = 100$ ) from very few data points (10 observed words on average). It would be desirable if topic models can exploit missing words (a lot more data than observed words) to render more nuanced latent semantics, so that pairs of sentences in the same domain can be differentiable.

On the other hand, LSA explicitly models missing words but not at the right level of emphasis. LSA finds another matrix  $\hat{X}$  (latent vectors) with rank  $K$  to approximate  $X$  using Singular Vector Decomposition ( $X \approx \hat{X} = U_K \Sigma_K V_K^T$ ), such that the Frobe-

	financial	sport	institution	$R_o$	$R_m$	$R_o - R_m$	$R_o - 0.01R_m$
$v_1$	1	0	0	<b>20</b>	<b>600</b>	-580	14
$v_2$	0.6	0	0.1	18	300	-282	<b>15</b>
$v_3$	0.2	0.3	0.2	5	100	<b>-95</b>	4

Table 1: Three possible latent vectors hypotheses for the definition of *bank#n#1*

nius norm of difference between the two matrices is minimized:

$$\sqrt{\sum_i \sum_j (\hat{X}_{ij} - X_{ij})^2} \quad (2)$$

In effect, LSA allows missing and observed words to equally impact the objective function. Given the inherent short length of the sentences, LSA (equation 2) allows for much more potential influence from missing words rather than observed words (99.9% cells are 0 in  $X$ ). Hence the contribution of the observed words is significantly diminished. Moreover, the true semantics of the concept definitions is actually related to some missing words, but such true semantics will not be favored by the objective function, since equation 2 allows for too strong an impact by  $\hat{X}_{ij} = 0$  for **any** missing word. Therefore the LSA model, in the context of short texts, is allowing missing words to have a significant “uncontrolled” impact on the model.

## 2.1 An Example

The three latent semantics profiles in table 1 illustrate our analysis for topic models and LSA. Assume there are three dimensions: financial, sports, institution. We use  $R_o^v$  to denote the sum of relatedness between latent vector  $v$  and all observed words; similarly,  $R_m^v$  is the sum of relatedness between the vector  $v$  and all missing words. The first latent vector (generated by topic models) is chosen by maximizing  $R_{obs} = 600$ . It suggests *bank#n#1* is only related to the *financial* dimension. The second latent vector (found by LSA) has the maximum value of  $R_{obs} - R_{miss} = 95$ , but obviously the latent vector is not related to *bank#n#1* at all. This is because LSA treats observed words and missing words equally the same, and due to the large number of missing words, the information of observed words is lost:  $R_{obs} - R_{miss} \approx -R_{miss}$ . The third vector is the ideal semantics profile, since it is also related to the *institution* dimension. It has a slightly smaller  $R_{obs}$  in comparison to the first vector, yet it has a substantially smaller  $R_{miss}$ .

In order to favor the ideal vector over other vectors, we simply need to adjust the objective func-

tion by assigning a smaller weight to  $R_{miss}$  such as:  $R_{obs} - 0.01 \times R_{miss}$ . Accordingly, we use weighted matrix factorization (Srebro and Jaakkola, 2003) to model missing words.

## 3 The Proposed Approach

### 3.1 Weighted Matrix Factorization

The weighted matrix factorization [WMF] approach is very similar to SVD, except that it allows for direct control on each matrix cell  $X_{ij}$ . The model factorizes the original matrix  $X$  into two matrices such that  $X \approx P^T Q$ , where  $P$  is a  $K \times M$  matrix, and  $Q$  is a  $K \times N$  matrix (figure 1).

The model parameters (vectors in  $P$  and  $Q$ ) are optimized by minimizing the objective function:

$$\sum_i \sum_j W_{ij} (P_{\cdot,i} \cdot Q_{\cdot,j} - X_{ij})^2 + \lambda \|P\|_2^2 + \lambda \|Q\|_2^2 \quad (3)$$

where  $\lambda$  is a free regularization factor, and the weight matrix  $W$  defines a weight for each cell in  $X$ .

Accordingly,  $P_{\cdot,i}$  is a  $K$ -dimension latent semantics vector profile for word  $w_i$ ; similarly,  $Q_{\cdot,j}$  is the  $K$ -dimension vector profile that represents the sentence  $s_j$ . Operations on these  $K$ -dimensional vectors have very intuitive semantic meanings:

- (1) the inner product of  $P_{\cdot,i}$  and  $Q_{\cdot,j}$  is used to approximate semantic relatedness of word  $w_i$  and sentence  $s_j$ :  $P_{\cdot,i} \cdot Q_{\cdot,j} \approx X_{ij}$ , as the shaded parts in Figure 1;
- (2) equation 3 explicitly requires a sentence should not be related to its missing words by forcing  $P_{\cdot,i} \cdot Q_{\cdot,j} = 0$  for missing words  $X_{ij} = 0$ .
- (3) we can compute the similarity of two sentences  $s_j$  and  $s_{j'}$  using the cosine similarity between  $Q_{\cdot,j}$ ,  $Q_{\cdot,j'}$ .

The latent vectors in  $P$  and  $Q$  are first randomly initialized, then can be computed iteratively by the following equations (derivation is omitted due to limited space, which can be found in (Srebro and Jaakkola, 2003)):

$$\begin{aligned} P_{\cdot,i} &= (Q\tilde{W}^{(i)}Q^T + \lambda I)^{-1} Q\tilde{W}^{(i)}X_{i\cdot}^T \\ Q_{\cdot,j} &= (P\tilde{W}^{(j)}P^T + \lambda I)^{-1} P\tilde{W}^{(j)}X_{\cdot j} \end{aligned} \quad (4)$$

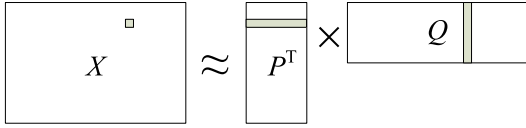


Figure 1: Matrix Factorization

where  $\tilde{W}^{(i)} = \text{diag}(W_{\cdot,i})$  is an  $M \times M$  diagonal matrix containing  $i$ th row of weight matrix  $W$ . Similarly,  $\tilde{W}^{(j)} = \text{diag}(W_{\cdot,j})$  is an  $N \times N$  diagonal matrix containing  $j$ th column of  $W$ .

### 3.2 Modeling Missing Words

It is straightforward to implement the idea in Section 2.1 (choosing a latent vector that maximizes  $R_{obs} - 0.01 \times R_{miss}$ ) in the WMF framework, by assigning a small weight for all the missing words and minimizing equation 3:

$$W_{i,j} = \begin{cases} 1, & \text{if } X_{ij} \neq 0 \\ w_m, & \text{if } X_{ij} = 0 \end{cases} \quad (5)$$

We refer to our model as Weighted Textual Matrix Factorization [WTMF].<sup>1</sup>

This solution is quite elegant: 1. it explicitly tells the model that in general all missing words should not be related to the sentence; 2. meanwhile latent semantics are mainly generalized based on observed words, and the model is not penalized too much ( $w_m$  is very small) when it is very confident that the sentence is highly related to a small subset of missing words based on their latent semantics profiles (*bank#n#1* definition sentence is related to its missing words *check loan*).

We adopt the same approach (assigning a small weight for some cells in WMF) proposed for recommender systems [RS] (Steck, 2010). In RS, an incomplete rating matrix  $R$  is proposed, where rows are users and columns are items. Typically, a user rates only some of the items, hence, the RS system needs to predict the missing ratings. Steck (2010) guesses a value for all the missing cells, and sets a small weight for those cells.

Compared to (Steck, 2010), we are facing a different problem and targeting a different goal. We have a full matrix  $X$  where missing words have a 0 value, while the missing ratings in RS are unavailable – the values are unknown, hence  $R$  is not complete. In the RS setting, they are interested in predicting individual ratings, while we are interested in the sentence

<sup>1</sup>An efficient way to compute equation 4 is proposed in (Steck, 2010).

semantics. More importantly, they do not have the sparsity issue (each user has rated over 100 items in the movie lens data<sup>2</sup>) and robust predictions can be made based on the observed ratings alone.

## 4 Evaluation for SS

We need to show the impact of our proposed model WTMF on the SS task. However we are faced with a problem, the lack of a suitable large evaluation set from which we can derive robust observations. The two data sets we know of for SS are: 1. human-rated sentence pair similarity data set (Li et al., 2006) [LI06]; 2. the Microsoft Research Paraphrase Corpus (Dolan et al., 2004) [MSR04]. The LI06 data set consists of 65 pairs of noun definitions selected from the Collin Cobuild Dictionary. A subset of 30 pairs is further selected by LI06 to render the similarity scores evenly distributed. While this is the ideal data set for SS, the small size makes it impossible for tuning SS algorithms or deriving significant performance conclusions.

On the other hand, the MSR04 data set comprises a much larger set of sentence pairs: 4,076 training and 1,725 test pairs. The ratings on the pairs are binary labels: similar/not similar. This is not a problem per se, however the issue is that it is very strict in its assignment of a positive label, for example the following sentence pair as cited in (Islam and Inkpen, 2008) is rated not semantically similar:

*Ballmer has been vocal in the past warning that Linux is a threat to Microsoft.*

*In the memo, Ballmer reiterated the open-source threat to Microsoft.*

We believe that the ratings on a data set for SS should accommodate variable degrees of similarity with various ratings, however such a large scale set does not exist yet. Therefore for purposes of evaluating our proposed approach we devise a new framework inspired by the LI06 data set in that it comprises concept definitions but on a large scale.

### 4.1 Concept Definition Retrieval

We define a new framework for evaluating SS and project it as a Concept Definition Retrieval (CDR) task where the data points are dictionary definitions. The intuition is that two definitions in different dic-

<sup>2</sup><http://www.grouplens.org/node/73>, with 1M data set being the most widely used.

tionaries referring to the same concept should be assigned large similarity. In this setting, we design the CDR task in a search engine style. The SS algorithm has access to all the definitions in WordNet (WN). Given an OntoNotes (ON) definition (Hovy et al., 2006), the SS algorithm should rank the equivalent WN definition as high as possible based on sentence similarity.

The manual mapping already exists for ON to WN. One ON definition can be mapped to several WN definitions. After preprocessing we obtain 13669 ON definitions mapped to 19655 WN definitions. The data set has the advantage of being very large and it doesn't require further human scrutiny.

After the SS model learns the co-occurrence of words from WN definitions, in the testing phase, given an ON definition  $d$ , the SS algorithm needs to identify the equivalent WN definitions by computing the similarity values between all WN definitions and the ON definition  $d$ , then sorting the values in decreasing order. Clearly, it is very difficult to rank the one correct definition as highest out of all WN definitions (110,000 in total), hence we use  $ATOP_d$ , *area under the TOPK $_d(k)$  recall curve for an ON definition  $d$* , to measure the performance. Basically, it is the ranking of the correct WN definition among all WN definitions. The higher a model is able to rank the correct WN definition, the better its performance.

Let  $N_d$  be the number of aligned WN definitions for the ON definition  $d$ , and  $N_d^k$  be the number of aligned WN definitions in the top-k list. Then with a normalized  $k \in [0,1]$ ,  $TOPK_d(k)$  and  $ATOP_d$  is defined as:

$$\begin{aligned} TOPK_d(k) &= N_d^k / N_d \\ ATOP_d &= \int_0^1 TOPK_d(k) dk \end{aligned} \quad (6)$$

$ATOP_d$  computes the normalized rank (in the range of  $[0, 1]$ ) of aligned WN definitions among all WN definitions, with value 0.5 being the random case, and 1 being ranked as most similar.

## 5 Experiments and Results

We evaluate WTMF on three data sets: 1. CDR data set using  $ATOP$  metric; 2. Human-rated Sentence Similarity data set [LI06] using Pearson and Spearman Correlation; 3. MSR Paraphrase corpus [MSR04] using accuracy.

The performance of WTMF on CDR is compared with (a) an Information Retrieval model (IR) that is based on surface word matching, (b) an n-gram model (N-gram) that captures phrase overlaps by returning the number of overlapping ngrams as the similarity score of two sentences, (c) LSA that uses `svds()` function in Matlab, and (d) LDA that uses Gibbs Sampling for inference (Griffiths and Steyvers, 2004). WTMF is also compared with all existing reported SS results on LI06 and MSR04 data sets, as well as LDA that is trained on the same data as WTMF. The similarity of two sentences is computed by cosine similarity (except N-gram). More details on each task will be explained in the subsections.

To eliminate randomness in statistical models (WTMF and LDA), all the reported results are averaged over 10 runs. We run 20 iterations for WTMF. And we run 5000 iterations for LDA; each LDA model is averaged over the last 10 Gibbs Sampling iterations to get more robust predictions.

The latent vector of a sentence is computed by: (1) using equation 4 in WTMF, or (2) summing up the latent vectors of all the constituent words weighted by  $X_{ij}$  in LSA and LDA, similar to the work reported in (Mihalcea et al., 2006). For LDA the latent vector of a word is computed by  $P(z|w)$ . It is worth noting that we could directly use the estimated topic distribution  $\theta_j$  to represent a sentence, however, as discussed the topic distribution has only non-zero values on one or two topics, leading to a low  $ATOP$  value around 0.8.

### 5.1 Corpus

The corpus we use comprises three dictionaries WN, ON, Wiktionary [Wik],<sup>3</sup> Brown corpus. For all dictionaries, we only keep the definitions without examples, and discard the mapping between sense ids and definitions. All definitions are simply treated as individual documents. We crawl Wik and remove the entries that are not tagged as noun, verb, adjective, or adverb, resulting in 220,000 entries. For the Brown corpus, each sentence is treated as a document in order to create more coherent co-occurrence values. All data is tokenized, pos-tagged<sup>4</sup>, and lem-

<sup>3</sup>[http://en.wiktionary.org/wiki/Wiktionary:Main\\_Page](http://en.wiktionary.org/wiki/Wiktionary:Main_Page)

<sup>4</sup><http://nlp.stanford.edu/software/tagger.shtml>



Models	Parameters	Dev	Test
1. IR	-	0.8578	0.8515
2. N-gram	-	0.8238	0.8171
3. LSA	-	0.8218	0.8143
4a. LDA	$\alpha = 0.1, \beta = 0.01$	$0.9466 \pm 0.0020$	$0.9427 \pm 0.0006$
4b. LDA	$\alpha = 0.05, \beta = 0.05$	$0.9506 \pm 0.0017$	$0.9470 \pm 0.0005$
5. WTMF	$w_m = 1, \lambda = 0$	$0.8273 \pm 0.0028$	$0.8273 \pm 0.0014$
6. WTMF	$w_m = 0, \lambda = 20$	$0.8745 \pm 0.0058$	$0.8645 \pm 0.0031$
7a. WTMF	$w_m = 0.01, \lambda = 20$	$0.9555 \pm 0.0015$	$0.9511 \pm 0.0003$
7b. WTMF	$w_m = 0.0005, \lambda = 20$	$0.9610 \pm 0.0011$	$0.9558 \pm 0.0004$

Table 2: ATOP Values of Models ( $K = 100$  for LSA/LDA/WTMF)

matized<sup>5</sup>. The importance of words in a sentence is estimated by the TF-IDF schema.

All the latent variable models (LSA, LDA, WTMF) are built on the same set of corpus: WN+Wik+Brown (393, 666 sentences and 4, 262, 026 words). Words that appear only once are removed. The test data is never used during training phrase.

## 5.2 Concept Definition Retrieval

Among the 13669 ON definitions, 1000 definitions are randomly selected as a development set (dev) for picking best parameters in the models, and the rest is used as a test set (test). The performance of each model is evaluated by the average  $ATOP_d$  value over the 12669 definitions (test). We use the subscript *set* in  $ATOP_{set}$  to denote the average of  $ATOP_d$  of a set of ON definitions, where  $d \in \{set\}$ . If all the words in an ON definition are not covered in the training data (WN+Wik+Br), then  $ATOP_d$  for this instance is set to 0.5.

To compute  $ATOP_d$  for an ON definition efficiently, we use the rank of the aligned WN definition among a random sample (size=1000) of WN definitions, to approximate its rank among all WN definitions. In practice, the difference between using 1000 samples and all data is tiny for  $ATOP_{test}$  ( $\pm 0.0001$ ), due to the large number of data points in CDR.

We mainly compare the performance of IR, N-gram, LSA, LDA, and WTMF models. Generally results are reported based on the last iteration. However, we observe that for model 6 in table 2, the best performance occurs at the first few iterations. Hence for that model we use the  $ATOP_{dev}$  to indicate when to stop.

<sup>5</sup><http://wn-similarity.sourceforge.net>, WordNet::QueryData

### 5.2.1 Results

Table 2 summarizes the ATOP values on the dev and test sets. All parameters are tuned based on the dev set. In LDA, we choose an optimal combination of  $\alpha$  and  $\beta$  from  $\{0.01, 0.05, 0.1, 0.5\}$ . In WTMF, we choose the best parameters of weight  $w_m$  for missing words and  $\lambda$  for regularization. We fix the dimension  $K = 100$ . Later in section 5.2.2, we will see that a larger value of  $K$  can further improve the performance.

WTMF that models missing words using a small weight (model 7b with  $w_m = 0.0005$ ) outperforms the second best model LDA by a large margin. This is because LDA only uses 10 observed words to infer a 100 dimension vector for a sentence, while WTMF takes advantage of much more missing words to learn more robust latent semantics vectors.

The IR model that works in word space achieves better ATOP scores than N-gram, although the idea of N-gram is commonly used in detecting paraphrases as well as machine translation. Applying TF-IDF for N-gram is better, but still the  $ATOP_{test}$  is not higher: 0.8467. The reason is words are enough to capture semantics for SS, while n-grams/phrases are used for a more fine-grained level of semantics.

We also present model 5 and 6 (both are WTMF), to show the impact of: 1. modeling missing words with equal weights as observed words ( $w_m = 1$ ) (LSA manner), and 2. not modeling missing words at all ( $w_m = 0$ ) (LDA manner) in the context of WTMF model. As expected, both model 5 and model 6 generate much worse results.

Both LDA and model 6 ignore missing words, with better  $ATOP_{test}$  scores achieved by LDA. This may be due to the different inference algorithms. Model 5 and LSA are comparable, where missing words are used with a large weight. Both of them yield low results. This confirms our assumption

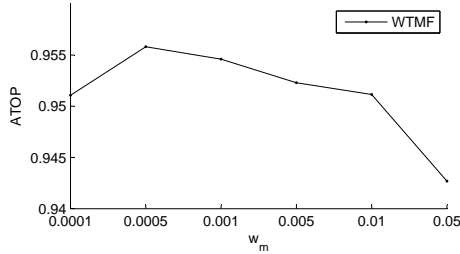


Figure 2: missing words weight  $w_m$  in WTMF

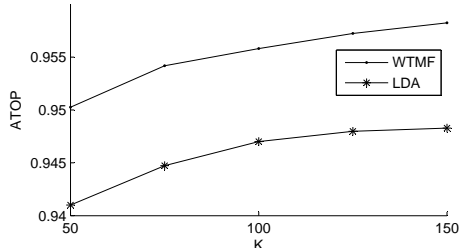


Figure 3: dimension  $K$  in WTMF and LDA

that allowing for equal impact of both observed and missing words is not the correct characterization of the semantic space.

### 5.2.2 Analysis

In these latent variable models, there are several essential parameters: weight of missing words  $w_m$ , and dimension  $K$ . Figure 2 and 3 analyze the impact of these parameters on  $ATOP_{test}$ .

Figure 2 shows the influence of  $w_m$  on  $ATOP_{test}$  values. The peak  $ATOP_{test}$  is around  $w_m = 0.0005$ , while other values of  $w_m$  (except  $w_m = 0.05$ ) also yield high ATOP values (better than LDA).

We also measure the influence of the dimension  $K = \{50, 75, 100, 125, 150\}$  on LDA and WTMF in Figure 3, where parameters for WTMF are  $w_m = 0.0005$ ,  $\lambda = 20$ , and for LDA are  $\alpha = 0.05$ ,  $\beta = 0.05$ . We can see WTMF consistently outperforms LDA by an ATOP value of 0.01 in each dimension. Although a larger  $K$  yields a better result, we still use a 100 due to computational complexity.

### 5.3 LI06: Human-rated Sentence Similarity

We also assess WTMF and LDA model on LI06 data set. We still use  $K = 100$ . As we can see in Figure 2, choosing the appropriate parameter  $w_m$  could boost the performance significantly. Since we do not have any tuning data for this task, we present Pearson’s correlation  $r$  for different values of  $w_m$  in Table 3. In addition, to demonstrate that  $w_m$  does not overfit the 30 data points, we also evaluate on

$w_m$	30 pairs		35 pairs	
	$r$	$\rho$	$r$	$\rho$
0.0005	0.8247	0.8440	0.4200	0.6006
0.001	0.8470	0.8636	0.4308	0.5985
0.005	0.8876	0.8966	<b>0.4638</b>	<b>0.5809</b>
0.01	<b>0.8984</b>	<b>0.9091</b>	0.4564	0.5450
0.05	0.8804	0.8812	0.4087	0.4766

Table 3: Different  $w_m$  of WTMF on LI06 ( $K = 100$ )

the other 35 pairs in LI06. Same as in (Tsatsaronis et al., 2010), we also include Spearman’s rank order correlation  $\rho$ , which is correlation of ranks of similarity values. Note that  $r$  and  $\rho$  are much lower for 35 pairs set, since most of the sentence pairs have a very low similarity (the average similarity value is 0.065 in 35 pairs set and 0.367 in 30 pairs set) and SS models need to identify the tiny difference among them, thereby rendering this set much harder to predict.

Using  $w_m = 0.01$  gives the best results on 30 pairs while on 35 pairs the peak values of  $r$  and  $\rho$  happens when  $w_m = 0.005$ . In general, the correlations in 30 pairs and in 35 pairs are consistent, which indicates  $w_m = 0.01$  or  $w_m = 0.005$  does not overfit the 30 pairs set.

Compared to CDR, LI06 data set has a strong preference for a larger  $w_m$ . This could be caused by different goals of the two tasks: CDR is evaluated by the rank of the most similar ones among all candidates, while the LI06 data set treats similar pairs and dissimilar pairs as equally important. Using a smaller  $w_m$  means the similarity score is computed mainly from semantics of the observed words. This benefits CDR, since it gives more accurate similarity scores for those similar pairs, but not so accurate for dissimilar pairs. In fact, from Figure 2 and Table 2 we see that  $w_m = 0.01$  also produces a very high  $ATOP_{test}$  value in CDR.

Table 4 shows the results of all current SS models with respect to the LI06 data set (30 pairs set). We cite their best performance for all reported results.

Once the correct  $w_m = 0.01$  is chosen, WTMF results in the best Pearson’s  $r$  and best Spearman’s  $\rho$  ( $w_m = 0.005$  yields the second best  $r$  and  $\rho$ ). Same as in CDR task, WTMF outperforms LDA by a large margin in both  $r$  and  $\rho$ . It indicates that the latent vectors induced by WTMF are able to not only identify same/similar sentences, but also identify the “correct” degree of dissimilar sentences.

Model	$r$	$\rho$
STASIS (Li et al., 2006)	0.8162	0.8126
(Liu et al., 2007)	0.841	0.8538
(Feng et al., 2008)	0.756	0.608
STS (Islam and Inkpen, 2008)	0.853	0.838
LSA (O’Shea et al., 2008)	0.8384	0.8714
Omiotis (Tsatsaronis et al., 2010)	0.856	0.8905
WSD-STIS (Ho et al., 2010)	0.864	0.8341
SPD-STIS (Ho et al., 2010)	0.895	0.9034
LDA ( $\alpha = 0.05, \beta = 0.05$ )	0.8422	0.8663
WTMF ( $w_m = 0.005, \lambda = 20$ )	0.8876	0.8966
WTMF ( $w_m = 0.01, \lambda = 20$ )	<b>0.8984</b>	<b>0.9091</b>

Table 4: Pearson’s correlation  $r$  and Spearman’s correlation  $\rho$  on LI06 30 pairs

Model	Accuracy
Random	51.3
LSA (Mihalcea et al., 2006)	68.4
full model (Mihalcea et al., 2006)	70.3
STS (Islam and Inkpen, 2008)	72.6
Omiotis (Tsatsaronis et al., 2010)	69.97
LDA ( $\alpha = 0.05, \beta = 0.05$ )	68.6
WTMF ( $w_m = 0.01, \lambda = 20$ )	71.51

Table 5: Performance on MSR04 test set

#### 5.4 MSR04: MSR Paraphrase Corpus

Finally, we briefly discuss results of applying WTMF on MSR04 data. We use the same parameter setting used for the LI06 evaluation setting since both sets are human-rated sentence pairs ( $\lambda = 20, w_m = 0.01, K = 100$ ). We use the training set of MSR04 data to select a threshold of sentence similarity for the binary label. Table 5 summarizes the accuracy of other SS models noted in the literature and evaluated on MSR04 test set.

Compared to previous SS work and LDA, WTMF has the second best accuracy. It suggests that WTMF is quite competitive in the paraphrase recognition task.

It is worth noting that the best system on MSR04, STS (Islam and Inkpen, 2008), has much lower correlations on LI06 data set. The second best system among previous work on LI06 uses Spearman correlation, Omiotis (Tsatsaronis et al., 2010), and it yields a much worse accuracy on MSR04. The other works do not evaluate on both data sets.

## 6 Related Work

Almost all current SS methods work in the high-dimensional word space, and rely heavily on word/sense similarity measures, which is knowledge based (Li et al., 2006; Feng et al., 2008; Ho et al., 2010; Tsatsaronis et al., 2010), corpus-based (Islam

and Inkpen, 2008) or hybrid (Mihalcea et al., 2006). Almost all of them are evaluated on LI06 data set. It is interesting to see that most works find word similarity measures, especially knowledge based ones, to be the most effective component, while other features do not work well (such as word order or syntactic information). Mihalcea et al. (2006) use LSA as a baseline, and O’Shea et al. (2008) train LSA on regular length documents. Both results are considerably lower than word similarity based methods. Hence, our work is the first to successfully approach SS in the latent space.

Although there has been work modeling latent semantics for short texts (tweets) in LDA, the focus has been on exploiting additional features in Twitter, hence restricted to Twitter data. Ramage et al. (2010) use tweet metadata (author, hashtag) as some supervised information to model tweets. Jin et al. (2011) use long similar documents (the article that is referred by a url in tweets) to help understand the tweet. In contrast, our approach relies solely on the information in the texts by modeling local missing words, and does not need any additional data, which renders our approach much more widely applicable.

## 7 Conclusions

We explicitly model missing words to alleviate the sparsity problem in modeling short texts. We also propose a new evaluation framework for sentence similarity that allows large scale tuning and testing. Experiment results on three data sets show that our model WTMF significantly outperforms existing methods. For future work, we would like to compare the text modeling performance of WTMF with LSA and LDA on regular length documents.

## Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

This research was funded by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), through the U.S. Army Research Lab. All statements of fact, opinion or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of IARPA, the ODNI or the U.S. Government.

## References

- David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3.
- William Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*.
- Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.
- Jin Feng, Yi-Ming Zhou, and Trevor Martin. 2008. Sentence similarity based on relevance. In *Proceedings of IPMU*.
- Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101.
- Chukfong Ho, Masrah Azrifah Azmi Murad, Rabiah Abdul Kadir, and Shyamala C. Doraisamy. 2010. Word sense disambiguation-based sentence similarity. In *Proceedings of the 23rd International Conference on Computational Linguistics*.
- Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*.
- Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.
- Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data*, 2.
- Ou Jin, Nathan N. Liu, Kai Zhao, Yong Yu, and Qiang Yang. 2011. Transferring topical knowledge from auxiliary long texts for short text clustering. In *Proceedings of the 20th ACM international conference on Information and knowledge management*.
- David Kauchak and Regina Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*.
- Thomas K Landauer, Peter W. Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. *Discourse Processes*, 25.
- Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*.
- Yuhua Li, David McLean, Zuhair A. Bandar, James D. O'Shea, and Keeley Crockett. 2006. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transaction on Knowledge and Data Engineering*, 18.
- Xiao-Ying Liu, Yi-Ming Zhou, and Ruo-Shi Zheng. 2007. Sentence similarity based on dynamic time warping. In *The International Conference on Semantic Computing*.
- Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence*.
- James O'Shea, Zuhair Bandar, Keeley Crockett, and David McLean. 2008. A comparative study of two short text semantic similarity measures. In *Proceedings of the Agent and Multi-Agent Systems: Technologies and Applications, Second KES International Symposium (KES-AMSTA)*.
- Daniel Ramage, Susan Dumais, and Dan Liebling. 2010. Characterizing microblogs with topic models. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*.
- Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning*.
- Harald Steck. 2010. Training and testing of recommender systems on data missing not at random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.
- George Tsatsaronis, Iraklis Varlamis, and Michalis Vazirgiannis. 2010. Text relatedness based on a word thesaurus. *Journal of Artificial Intelligence Research*, 37.
- Liang Zhou, Chin-Yew Lin, Dragos Stefan Munteanu, and Eduard Hovy. 2006. Paraeval: Using paraphrases to evaluate summaries automatically. In *Proceedings of Human Language Technology Conference of the North American Chapter of the ACL*.

# Improving Word Representations via Global Context and Multiple Word Prototypes

Eric H. Huang, Richard Socher\*, Christopher D. Manning, Andrew Y. Ng  
Computer Science Department, Stanford University, Stanford, CA 94305, USA  
{ehhuang, manning, ang}@stanford.edu, \*richard@socher.org

## Abstract

Unsupervised word representations are very useful in NLP tasks both as inputs to learning algorithms and as extra word features in NLP systems. However, most of these models are built with only local context and one representation per word. This is problematic because words are often polysemous and global context can also provide useful information for learning word meanings. We present a new neural network architecture which 1) learns word embeddings that better capture the semantics of words by incorporating both local and global document context, and 2) accounts for homonymy and polysemy by learning multiple embeddings per word. We introduce a new dataset with human judgments on pairs of words in sentential context, and evaluate our model on it, showing that our model outperforms competitive baselines and other neural language models.<sup>1</sup>

## 1 Introduction

Vector-space models (VSM) represent word meanings with vectors that capture semantic and syntactic information of words. These representations can be used to induce similarity measures by computing distances between the vectors, leading to many useful applications, such as information retrieval (Manning et al., 2008), document classification (Sebastiani, 2002) and question answering (Tellex et al., 2003).

<sup>1</sup>The dataset and word vectors can be downloaded at <http://ai.stanford.edu/~ehhuang/>.

Despite their usefulness, most VSMS share a common problem that each word is only represented with one vector, which clearly fails to capture homonymy and polysemy. Reisinger and Mooney (2010b) introduced a multi-prototype VSM where word sense discrimination is first applied by clustering contexts, and then prototypes are built using the contexts of the sense-labeled words. However, in order to cluster accurately, it is important to capture both the syntax and semantics of words. While many approaches use local contexts to disambiguate word meaning, global contexts can also provide useful topical information (Ng and Zelle, 1997). Several studies in psychology have also shown that global context can help language comprehension (Hess et al., 1995) and acquisition (Li et al., 2000).

We introduce a new neural-network-based language model that distinguishes and uses both local and global context via a joint training objective. The model learns word representations that better capture the semantics of words, while still keeping syntactic information. These improved representations can be used to represent contexts for clustering word instances, which is used in the multi-prototype version of our model that accounts for words with multiple senses.

We evaluate our new model on the standard WordSim-353 (Finkelstein et al., 2001) dataset that includes human similarity judgments on pairs of words, showing that combining both local and global context outperforms using only local or global context alone, and is competitive with state-of-the-art methods. However, one limitation of this evaluation is that the human judgments are on pairs

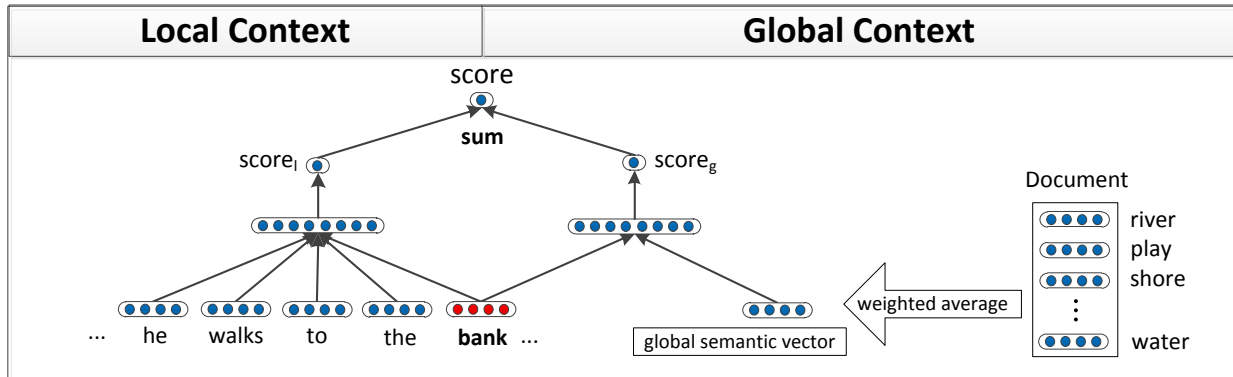


Figure 1: An overview of our neural language model. The model makes use of both local and global context to compute a score that should be large for the actual next word (*bank* in the example), compared to the score for other words. When word meaning is still ambiguous given local context, information in global context can help disambiguation.

of words presented in *isolation*, ignoring meaning variations in context. Since word interpretation in context is important especially for homonymous and polysemous words, we introduce a new dataset with human judgments on similarity between pairs of words in sentential context. To capture interesting word pairs, we sample different senses of words using WordNet (Miller, 1995). The dataset includes verbs and adjectives, in addition to nouns. We show that our multi-prototype model improves upon the single-prototype version and outperforms other neural language models and baselines on this dataset.

## 2 Global Context-Aware Neural Language Model

In this section, we describe the training objective of our model, followed by a description of the neural network architecture, ending with a brief description of our model’s training method.

### 2.1 Training Objective

Our model jointly learns word representations while learning to discriminate the next word given a short word sequence (local context) and the document (global context) in which the word sequence occurs. Because our goal is to learn useful word representations and not the *probability* of the next word given previous words (which prohibits looking ahead), our model can utilize the entire document to provide

global context.

Given a word sequence  $s$  and document  $d$  in which the sequence occurs, our goal is to discriminate the correct last word in  $s$  from other random words. We compute scores  $g(s, d)$  and  $g(s^w, d)$  where  $s^w$  is  $s$  with the last word replaced by word  $w$ , and  $g(\cdot, \cdot)$  is the scoring function that represents the neural networks used. We want  $g(s, d)$  to be larger than  $g(s^w, d)$  by a margin of 1, for any other word  $w$  in the vocabulary, which corresponds to the training objective of minimizing the ranking loss for each  $(s, d)$  found in the corpus:

$$C_{s,d} = \sum_{w \in V} \max(0, 1 - g(s, d) + g(s^w, d)) \quad (1)$$

Collobert and Weston (2008) showed that this ranking approach can produce good word embeddings that are useful in several NLP tasks, and allows much faster training of the model compared to optimizing log-likelihood of the next word.

### 2.2 Neural Network Architecture

We define two scoring components that contribute to the final score of a (word sequence, document) pair. The scoring components are computed by two neural networks, one capturing local context and the other global context, as shown in Figure 1. We now describe how each scoring component is computed.

The score of local context uses the local word sequence  $s$ . We first represent the word sequence  $s$  as

an ordered list of vectors  $x = (x_1, x_2, \dots, x_m)$  where  $x_i$  is the embedding of word  $i$  in the sequence, which is a column in the embedding matrix  $L \in \mathbb{R}^{n \times |V|}$  where  $|V|$  denotes the size of the vocabulary. The columns of this embedding matrix  $L$  are the word vectors and will be learned and updated during training. To compute the score of local context,  $\text{score}_l$ , we use a neural network with one hidden layer:

$$a_1 = f(W_1[x_1; x_2; \dots; x_m] + b_1) \quad (2)$$

$$\text{score}_l = W_2 a_1 + b_2 \quad (3)$$

where  $[x_1; x_2; \dots; x_m]$  is the concatenation of the  $m$  word embeddings representing sequence  $s$ ,  $f$  is an element-wise activation function such as  $\tanh$ ,  $a_1 \in \mathbb{R}^{h \times 1}$  is the activation of the hidden layer with  $h$  hidden nodes,  $W_1 \in \mathbb{R}^{h \times (mn)}$  and  $W_2 \in \mathbb{R}^{1 \times h}$  are respectively the first and second layer weights of the neural network, and  $b_1, b_2$  are the biases of each layer.

For the score of the global context, we represent the document also as an ordered list of word embeddings,  $d = (d_1, d_2, \dots, d_k)$ . We first compute the weighted average of all word vectors in the document:

$$c = \frac{\sum_{i=1}^k w(t_i) d_i}{\sum_{i=1}^k w(t_i)} \quad (4)$$

where  $w(\cdot)$  can be any weighting function that captures the importance of word  $t_i$  in the document. We use idf-weighting as the weighting function.

We use a two-layer neural network to compute the global context score,  $\text{score}_g$ , similar to the above:

$$a_1^{(g)} = f(W_1^{(g)}[c; x_m] + b_1^{(g)}) \quad (5)$$

$$\text{score}_g = W_2^{(g)} a_1^{(g)} + b_2^{(g)} \quad (6)$$

where  $[c; x_m]$  is the concatenation of the weighted average document vector and the vector of the last word in  $s$ ,  $a_1^{(g)} \in \mathbb{R}^{h^{(g)} \times 1}$  is the activation of the hidden layer with  $h^{(g)}$  hidden nodes,  $W_1^{(g)} \in \mathbb{R}^{h^{(g)} \times (2n)}$  and  $W_2^{(g)} \in \mathbb{R}^{1 \times h^{(g)}}$  are respectively the first and second layer weights of the neural network, and  $b_1^{(g)}, b_2^{(g)}$  are the biases of each layer. Note that instead of using the document where the sequence occurs, we can also specify a fixed  $k > m$  that captures larger context.

The final score is the sum of the two scores:

$$\text{score} = \text{score}_l + \text{score}_g \quad (7)$$

The local score preserves word order and syntactic information, while the global score uses a weighted average which is similar to bag-of-words features, capturing more of the semantics and topics of the document. Note that Collobert and Weston (2008)'s language model corresponds to the network using only local context.

### 2.3 Learning

Following Collobert and Weston (2008), we sample the gradient of the objective by randomly choosing a word from the dictionary as a *corrupt* example for each sequence-document pair,  $(s, d)$ , and take the derivative of the ranking loss with respect to the parameters: weights of the neural network and the embedding matrix  $L$ . These weights are updated via backpropagation. The embedding matrix  $L$  is the word representations. We found that word embeddings move to good positions in the vector space faster when using mini-batch L-BFGS (Liu and Nocedal, 1989) with 1000 pairs of good and corrupt examples per batch for training, compared to stochastic gradient descent.

### 3 Multi-Prototype Neural Language Model

Despite distributional similarity models' successful applications in various NLP tasks, one major limitation common to most of these models is that they assume only one representation for each word. This single-prototype representation is problematic because many words have multiple meanings, which can be wildly different. Using one representation simply cannot capture the different meanings. Moreover, using all contexts of a homonymous or polysemous word to build a single prototype could hurt the representation, which cannot represent any one of the meanings well as it is influenced by all meanings of the word.

Instead of using only one representation per word, Reisinger and Mooney (2010b) proposed the multi-prototype approach for vector-space models, which uses multiple representations to capture different senses and usages of a word. We show how our

model can readily adopt the multi-prototype approach. We present a way to use our learned single-prototype embeddings to represent each context window, which can then be used by clustering to perform word sense discrimination (Schütze, 1998).

In order to learn multiple prototypes, we first gather the fixed-sized context windows of all occurrences of a word (we use 5 words before and after the word occurrence). Each context is represented by a weighted average of the context words’ vectors, where again, we use idf-weighting as the weighting function, similar to the document context representation described in Section 2.2. We then use spherical k-means to cluster these context representations, which has been shown to model semantic relations well (Dhillon and Modha, 2001). Finally, each word occurrence in the corpus is re-labeled to its associated cluster and is used to train the word representation for that cluster.

Similarity between a pair of words  $(w, w')$  using the multi-prototype approach can be computed with or without context, as defined by Reisinger and Mooney (2010b):

$$\text{AvgSimC}(w, w') = \frac{1}{K^2} \sum_{i=1}^k \sum_{j=1}^k p(c, w, i) p(c', w', j) d(\mu_i(w), \mu_j(w')) \quad (8)$$

where  $p(c, w, i)$  is the likelihood that word  $w$  is in its cluster  $i$  given context  $c$ ,  $\mu_i(w)$  is the vector representing the  $i$ -th cluster centroid of  $w$ , and  $d(v, v')$  is a function computing similarity between two vectors, which can be any of the distance functions presented by Curran (2004). The similarity measure can be computed in absence of context by assuming uniform  $p(c, w, i)$  over  $i$ .

## 4 Experiments

In this section, we first present a qualitative analysis comparing the nearest neighbors of our model’s embeddings with those of others, showing our embeddings better capture the semantics of words, with the use of global context. Our model also improves the correlation with human judgments on a word similarity task. Because word interpretation in context is

important, we introduce a new dataset with human judgments on similarity of pairs of words in sentential context. Finally, we show that our model outperforms other methods on this dataset and also that the multi-prototype approach improves over the single-prototype approach.

We chose Wikipedia as the corpus to train all models because of its wide range of topics and word usages, and its clean organization of document by topic. We used the April 2010 snapshot of the Wikipedia corpus (Shaoul and Westbury, 2010), with a total of about 2 million articles and 990 million tokens. We use a dictionary of the 30,000 most frequent words in Wikipedia, converted to lower case. In preprocessing, we keep the frequent numbers intact and replace each digit of the uncommon numbers to “DG” so as to preserve information such as it being a year (e.g. “DGDGDGDG”). The converted numbers that are rare are mapped to a NUMBER token. Other rare words not in the dictionary are mapped to an UNKNOWN token.

For all experiments, our models use 50-dimensional embeddings. We use 10-word windows of text as the local context, 100 hidden units, and no weight regularization for both neural networks. For multi-prototype variants, we fix the number of prototypes to be 10.

### 4.1 Qualitative Evaluations

In order to show that our model learns more semantic word representations with global context, we give the nearest neighbors of our single-prototype model versus C&W’s, which only uses local context. The nearest neighbors of a word are computed by comparing the cosine similarity between the center word and all other words in the dictionary. Table 1 shows the nearest neighbors of some words. The nearest neighbors of “market” that C&W’s embeddings give are more constrained by the syntactic constraint that words in plural form are only close to other words in plural form, whereas our model captures that the singular and plural forms of a word are similar in meaning. Other examples show that our model induces nearest neighbors that better capture semantics.

Table 2 shows the nearest neighbors of our model using the multi-prototype approach. We see that the clustering is able to group contexts of different



Center Word	C&W	Our Model
markets	firms, industries, stores	market, firms, businesses
American	Australian, Indian, Italian	U.S., Canadian, African
illegal	alleged, overseas, banned	harmful, prohibited, convicted

Table 1: Nearest neighbors of words based on cosine similarity. Our model is less constrained by syntax and is more semantic.

Center Word	Nearest Neighbors
bank_1	corporation, insurance, company
bank_2	shore, coast, direction
star_1	movie, film, radio
star_2	galaxy, planet, moon
cell_1	telephone, smart, phone
cell_2	pathology, molecular, physiology
left_1	close, leave, live
left_2	top, round, right

Table 2: Nearest neighbors of word embeddings learned by our model using the multi-prototype approach based on cosine similarity. The clustering is able to find the different meanings, usages, and parts of speech of the words.

meanings of a word into separate groups, allowing our model to learn multiple meaningful representations of a word.

## 4.2 WordSim-353

A standard dataset for evaluating vector-space models is the WordSim-353 dataset (Finkelstein et al., 2001), which consists of 353 pairs of nouns. Each pair is presented without context and associated with 13 to 16 human judgments on similarity and relatedness on a scale from 0 to 10. For example, (cup,drink) received an average score of 7.25, while (cup,substance) received an average score of 1.92.

Table 3 shows our results compared to previous methods, including C&W’s language model and the hierarchical log-bilinear (HLBL) model (Mnih and Hinton, 2008), which is a probabilistic, linear neural model. We downloaded these embeddings from Turian et al. (2010). These embeddings were trained on the smaller corpus RCV1 that contains one year of Reuters English newswire, and show similar correlations on the dataset. We report the result of

Model	Corpus	$\rho \times 100$
Our Model-g	Wiki.	22.8
C&W	RCV1	29.5
HLBL	RCV1	33.2
C&W*	Wiki.	49.8
C&W	Wiki.	55.3
Our Model	Wiki.	64.2
Our Model*	Wiki.	71.3
Pruned <i>tf-idf</i>	Wiki.	73.4
ESA	Wiki.	75
Tiered Pruned <i>tf-idf</i>	Wiki.	76.9

Table 3: Spearman’s  $\rho$  correlation on WordSim-353, showing our model’s improvement over previous neural models for learning word embeddings. C&W\* is the word embeddings trained and provided by C&W. Our Model\* is trained without stop words, while Our Model-g uses only global context. Pruned *tf-idf* (Reisinger and Mooney, 2010b) and ESA (Gabrilovich and Markovitch, 2007) are also included.

our re-implementation of C&W’s model trained on Wikipedia, showing the large effect of using a different corpus.

Our model is able to learn more semantic word embeddings and noticeably improves upon C&W’s model. Note that our model achieves higher correlation (64.2) than either using local context alone (C&W: 55.3) or using global context alone (Our Model-g: 22.8). We also found that correlation can be further improved by removing stop words (71.3). Thus, each window of text (training example) contains more information but still preserves some syntactic information as the words are still ordered in the local context.

## 4.3 New Dataset: Word Similarity in Context

The many previous datasets that associate human judgments on similarity between pairs of words, such as WordSim-353, MC (Miller and Charles, 1991) and RG (Rubenstein and Goodenough, 1965), have helped to advance the development of vector-space models. However, common to all datasets is that similarity scores are given to pairs of words in *isolation*. This is problematic because the meanings of homonymous and polysemous words depend highly on the words’ contexts. For example, in the two phrases, “he swings the baseball *bat*” and “the

Word 1	Word 2
Located downtown along the east <b>bank</b> of the Des Moines River ...	This is the basis of all <b>money</b> laundering , a track record of depositing clean money before slipping through dirty money ...
Inside the ruins , there are <b>bats</b> and a bowl with Pokeys that fills with sand over the course of the race , and the music changes somewhat while inside ...	An aggressive lower order batsman who usually <b>bats</b> at No. 11 , Muralitharan is known for his tendency to back away to leg and slog ...
An example of legacy <b>left</b> in the Mideast from these nobles is the Krak des Chevaliers ' enlargement by the Counts of Tripoli and Toulouse ...	... one should not adhere to a particular explanation , only in such measure as to be ready to <b>abandon</b> it if it be proved with certainty to be false ...
... and Andy 's getting ready to <b>pack</b> his bags and head up to Los Angeles tomorrow to get ready to fly back home on Thursday	... she encounters Ben ( Duane Jones ) , who arrives in a pickup truck and defends the house against another <b>pack</b> of zombies ...
In <b>practice</b> , there is an unknown phase delay between the transmitter and receiver that must be compensated by " synchronization " of the receivers local oscillator	... but Gilbert did not believe that she was dedicated enough , and when she missed a <b>rehearsal</b> , she was dismissed ...

Table 4: Example pairs from our new dataset. Note that words in a pair can be the same word and have different parts of speech.

*bat* flies”, *bat* has completely different meanings. It is unclear how this variation in meaning is accounted for in human judgments of words presented without context.

One of the main contributions of this paper is the creation of a new dataset that addresses this issue. The dataset has three interesting characteristics: 1) human judgments are on pairs of words presented in sentential context, 2) word pairs and their contexts are chosen to reflect interesting variations in meanings of homonymous and polysemous words, and 3) verbs and adjectives are present in addition to nouns. We now describe our methodology in constructing the dataset.

#### 4.3.1 Dataset Construction

Our procedure of constructing the dataset consists of three steps: 1) select a list a words, 2) for each word, select another word to form a pair, 3) for each word in a pair, find a sentential context. We now describe each step in detail.

In step 1, in order to make sure we select a diverse list of words, we consider three attributes of a word: frequency in a corpus, number of parts of speech, and number of synsets according to WordNet. For frequency, we divide words into three groups, top 2,000 most frequent, between 2,000 and 5,000, and between 5,000 to 10,000 based on occurrences in Wikipedia. For number of parts of speech, we group words based on their number of possible parts of

speech (noun, verb or adjective), from 1 to 3. We also group words by their number of synsets: [0,5], [6,10], [11, 20], and [20, max]. Finally, we sample at most 15 words from each combination in the Cartesian product of the above groupings.

In step 2, for each of the words selected in step 1, we want to choose the other word so that the pair captures an interesting relationship. Similar to Manandhar et al. (2010), we use WordNet to first randomly select one synset of the first word, we then construct a set of words in various relations to the first word’s chosen synset, including hypernyms, hyponyms, holonyms, meronyms and attributes. We randomly select a word from this set of words as the second word in the pair. We try to repeat the above twice to generate two pairs for each word. In addition, for words with more than five synsets, we allow the second word to be the same as the first, but with different synsets. We end up with pairs of words as well as the one chosen synset for each word in the pairs.

In step 3, we aim to extract a sentence from Wikipedia for each word, which contains the word and corresponds to a usage of the chosen synset. We first find all sentences in which the word occurs. We then POS tag<sup>2</sup> these sentences and filter out those that do not match the chosen POS. To find the

<sup>2</sup>We used the MaxEnt Treebank POS tagger in the python nltk library.

Model	$\rho \times 100$
C&W-S	57.0
Our Model-S	58.6
Our Model-M AvgSim	62.8
Our Model-M AvgSimC	<b>65.7</b>
<i>tf-idf</i> -S	26.3
Pruned <i>tf-idf</i> -S	62.5
Pruned <i>tf-idf</i> -M AvgSim	60.4
Pruned <i>tf-idf</i> -M AvgSimC	60.5

Table 5: Spearman’s  $\rho$  correlation on our new dataset. Our Model-S uses the single-prototype approach, while Our Model-M uses the multi-prototype approach. AvgSim calculates similarity with each prototype contributing equally, while AvgSimC weighs the prototypes according to probability of the word belonging to that prototype’s cluster.

word usages that correspond to the chosen synset, we first construct a set of related words of the chosen synset, including hypernyms, hyponyms, holonyms, meronyms and attributes. Using this set of related words, we filter out a sentence if the document in which the sentence appears does not include one of the related words. Finally, we randomly select one sentence from those that are left.

Table 4 shows some examples from the dataset. Note that the dataset also includes pairs of the same word. Single-prototype models would give the max similarity score for those pairs, which can be problematic depending on the words’ contexts. This dataset requires models to examine context when determining word meaning.

Using Amazon Mechanical Turk, we collected 10 human similarity ratings for each pair, as Snow et al. (2008) found that 10 non-expert annotators can achieve very close inter-annotator agreement with expert raters. To ensure worker quality, we only allowed workers with over 95% approval rate to work on our task. Furthermore, we discarded all ratings by a worker if he/she entered scores out of the accepted range or missed a rating, signaling low-quality work.

We obtained a total of 2,003 word pairs and their sentential contexts. The word pairs consist of 1,712 unique words. Of the 2,003 word pairs, 1328 are noun-noun pairs, 399 verb-verb, 140 verb-noun, 97 adjective-adjective, 30 noun-adjective, and 9 verb-adjective. 241 pairs are same-word pairs.

### 4.3.2 Evaluations on Word Similarity in Context

For evaluation, we also compute Spearman correlation between a model’s computed similarity scores and human judgments. Table 5 compares different models’ results on this dataset. We compare against the following baselines: *tf-idf* represents words in a word-word matrix capturing co-occurrence counts in all 10-word context windows. Reisinger and Mooney (2010b) found pruning the low-value *tf-idf* features helps performance. We report the result of this pruning technique after tuning the threshold value on this dataset, removing all but the top 200 features in each word vector. We tried the same multi-prototype approach and used spherical k-means<sup>3</sup> to cluster the contexts using *tf-idf* representations, but obtained lower numbers than single-prototype (55.4 with AvgSimC). We then tried using pruned *tf-idf* representations on contexts with our clustering assignments (included in Table 5), but still got results worse than the single-prototype version of the pruned *tf-idf* model (60.5 with AvgSimC). This suggests that the pruned *tf-idf* representations might be more susceptible to noise or mistakes in context clustering.

By utilizing global context, our model outperforms C&W’s vectors and the above baselines on this dataset. With multiple representations per word, we show that the multi-prototype approach can improve over the single-prototype version without using context (62.8 vs. 58.6). Moreover, using AvgSimC<sup>4</sup> which takes contexts into account, the multi-prototype model obtains the best performance (65.7).

## 5 Related Work

Neural language models (Bengio et al., 2003; Mnih and Hinton, 2007; Collobert and Weston, 2008; Schwenk and Gauvain, 2002; Emami et al., 2003) have been shown to be very powerful at language modeling, a task where models are asked to accurately predict the next word given previously seen words. By using distributed representations of

<sup>3</sup>We first tried movMF as in Reisinger and Mooney (2010b), but were unable to get decent results (only 31.5).

<sup>4</sup>probability of being in a cluster is calculated as the inverse of the distance to the cluster centroid.

words which model words' similarity, this type of models addresses the data sparseness problem that  $n$ -gram models encounter when large contexts are used. Most of these models used relative local contexts of between 2 to 10 words. Schwenk and Gauvain (2002) tried to incorporate larger context by combining partial parses of past word sequences and a neural language model. They used up to 3 previous head words and showed increased performance on language modeling. Our model uses a similar neural network architecture as these models and uses the ranking-loss training objective proposed by Collobert and Weston (2008), but introduces a new way to combine local and global context to train word embeddings.

Besides language modeling, word embeddings induced by neural language models have been useful in chunking, NER (Turian et al., 2010), parsing (Socher et al., 2011b), sentiment analysis (Socher et al., 2011c) and paraphrase detection (Socher et al., 2011a). However, they have not been directly evaluated on word similarity tasks, which are important for tasks such as information retrieval and summarization. Our experiments show that our word embeddings are competitive in word similarity tasks.

Most of the previous vector-space models use a single vector to represent a word even though many words have multiple meanings. The multi-prototype approach has been widely studied in models of categorization in psychology (Rosseel, 2002; Griffiths et al., 2009), while Schütze (1998) used clustering of contexts to perform word sense discrimination. Reisinger and Mooney (2010b) combined the two approaches and applied them to vector-space models, which was further improved in Reisinger and Mooney (2010a). Two other recent papers (Dhillon et al., 2011; Reddy et al., 2011) present models for constructing word representations that deal with context. It would be interesting to evaluate those models on our new dataset.

Many datasets with human similarity ratings on pairs of words, such as WordSim-353 (Finkelstein et al., 2001), MC (Miller and Charles, 1991) and RG (Rubenstein and Goodenough, 1965), have been widely used to evaluate vector-space models. Motivated to evaluate composition models, Mitchell and Lapata (2008) introduced a dataset where an intransitive verb, presented with a subject noun, is com-

pared to another verb chosen to be either similar or dissimilar to the intransitive verb in context. The context is short, with only one word, and only verbs are compared. Erk and Padó (2008), Thater et al. (2011) and Dinu and Lapata (2010) evaluated word similarity in context with a modified task where systems are to rerank gold-standard paraphrase candidates given the SemEval 2007 Lexical Substitution Task dataset. This task only indirectly evaluates similarity as only reranking of already similar words are evaluated.

## 6 Conclusion

We presented a new neural network architecture that learns more semantic word representations by using both local and global context in learning. These learned word embeddings can be used to represent word contexts as low-dimensional weighted average vectors, which are then clustered to form different meaning groups and used to learn multi-prototype vectors. We introduced a new dataset with human judgments on similarity between pairs of words in context, so as to evaluate model's abilities to capture homonymy and polysemy of words in context. Our new multi-prototype neural language model outperforms previous neural models and competitive baselines on this new dataset.

## Acknowledgments

The authors gratefully acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181, and the DARPA Deep Learning program under contract number FA8650-10-C-7020. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, AFRL, or the US government.

## References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, Christian Jauvin, Jaz K, Thomas Hofmann, Tomaso Poggio, and John Shawe-taylor. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.
- James Richard Curran. 2004. From distributional to semantic similarity. Technical report.
- Inderjit S. Dhillon and Dharmendra S. Modha. 2001. Concept decompositions for large sparse text data using clustering. *Mach. Learn.*, 42:143–175, January.
- Paramveer S. Dhillon, Dean Foster, and Lyle Ungar. 2011. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24.
- Georgiana Dinu and Mirella Lapata. 2010. Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1162–1172, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Ahmad Emami, Peng Xu, and Frederick Jelinek. 2003. Using a connectionist model in a syntactical based language model. In *Acoustics, Speech, and Signal Processing*, pages 372–375.
- Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 897–906, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppín. 2001. Placing search in context: the concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 406–414, New York, NY, USA. ACM.
- Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on Artificial intelligence*, IJCAI'07, pages 1606–1611, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Thomas L Griffiths, Kevin R Canini, Adam N Sanborn, and Daniel J Navarro. 2009. Unifying rational models of categorization via the hierarchical dirichlet process. *Brain*, page 323328.
- David J Hess, Donald J Foss, and Patrick Carroll. 1995. Effects of global and local context on lexical processing during language comprehension. *Journal of Experimental Psychology: General*, 124(1):62–82.
- Ping Li, Curt Burgess, and Kevin Lund. 2000. The acquisition of word meaning through global lexical co-occurrences.
- D. C. Liu and J. Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Math. Program.*, 45(3):503–528, December.
- Suresh Manandhar, Ioannis P Klapaftis, Dmitriy Dligach, and Sameer S Pradhan. 2010. Semeval-2010 task 14: Word sense induction & disambiguation. *Word Journal Of The International Linguistic Association*, (July):63–68.
- Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.
- George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language & Cognitive Processes*, 6(1):1–28.
- George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38:39–41.
- Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *In Proceedings of ACL-08: HLT*, pages 236–244.
- Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 641–648, New York, NY, USA. ACM.
- Andriy Mnih and Geoffrey Hinton. 2008. A scalable hierarchical distributed language model. In *In NIPS*.
- Ht Ng and J Zelle. 1997. Corpus-based approaches to semantic interpretation in natural language processing. *AI Magazine*, 18(4):45–64.
- Siva Reddy, Ioannis Klapaftis, Diana McCarthy, and Suresh Manandhar. 2011. Dynamic and static prototype vectors for semantic composition. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 705–713, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Joseph Reisinger and Raymond Mooney. 2010a. A mixture model with sharing for lexical semantics. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1173–1182, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Joseph Reisinger and Raymond J. Mooney. 2010b. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 109–117, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Yves Rosseel. 2002. Mixture models of categorization. *Journal of Mathematical Psychology*, 46:178–210.

- Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM*, 8:627–633, October.
- Hinrich Schütze. 1998. Automatic word sense discrimination. *Journal of Computational Linguistics*, 24:97–123.
- Holger Schwenk and Jean-luc Gauvain. 2002. Connectionist language modeling for large vocabulary continuous speech recognition. In *International Conference on Acoustics, Speech and Signal Processing*, pages 765–768.
- Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34:1–47, March.
- Cyrus Shaoul and Chris Westbury. 2010. The westbury lab wikipedia corpus.
- Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP ’08*, pages 254–263, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*.
- Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. 2011b. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 26th International Conference on Machine Learning (ICML)*.
- Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011c. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Stefanie Tellex, Boris Katz, Jimmy Lin, Aaron Fernandes, and Gregory Marton. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Search and Development in Information Retrieval*, pages 41–47. ACM Press.
- Stefan Thater, Hagen Fürstenaу, and Manfred Pinkal. 2011. Word meaning in context: a simple and effective vector model. In *Proceedings of the 5th International Joint Conference on Natural Language Processing, IJCNLP ’11*.
- Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL ’10*, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics.

# Exploiting Social Information in Grounded Language Learning via Grammatical Reductions

**Mark Johnson**

Department of Computing  
Macquarie University  
Sydney, Australia

Mark.Johnson@MQ.edu.au

**Katherine Demuth**

Department of Linguistics  
Macquarie University  
Sydney, Australia

Katherine.Demuth@MQ.edu.au

**Michael Frank**

Department of Psychology  
Stanford University  
Stanford, California

mcf Frank@Stanford.edu

## Abstract

This paper uses an unsupervised model of grounded language acquisition to study the role that social cues play in language acquisition. The input to the model consists of (orthographically transcribed) child-directed utterances accompanied by the set of objects present in the non-linguistic context. Each object is annotated by *social cues*, indicating e.g., whether the caregiver is looking at or touching the object. We show how to model the task of inferring which objects are being talked about (and which words refer to which objects) as standard grammatical inference, and describe PCFG-based *unigram* models and adaptor grammar-based *collocation* models for the task. Exploiting social cues improves the performance of all models. Our models learn the relative importance of each social cue jointly with word-object mappings and collocation structure, consistent with the idea that children could discover the importance of particular social information sources during word learning.

## 1 Introduction

From learning sounds to learning the meanings of words, social interactions are extremely important for children's early language acquisition (Baldwin, 1993; Kuhl et al., 2003). For example, children who engage in more joint attention (e.g. looking at particular objects together) with caregivers tend to learn words faster (Carpenter et al., 1998). Yet computational or formal models of social interaction are rare, and those that exist have rarely gone beyond the stage of cue-weighting models. In order to study the role that *social cues* play in language acquisition, this paper presents a structured statistical model of

grounded learning that learns a mapping between words and objects from a corpus of child-directed utterances in a completely unsupervised fashion. It exploits five different *social cues*, which indicate which object (if any) the child is looking at, which object the child is touching, etc. Our models learn the salience of each social cue in establishing reference, relative to their co-occurrence with objects that are not being referred to. Thus, this work is consistent with a view of language acquisition in which children *learn to learn*, discovering organizing principles for how language is organized and used socially (Baldwin, 1993; Hollich et al., 2000; Smith et al., 2002).

We reduce the grounded learning task to a grammatical inference problem (Johnson et al., 2010; Börschinger et al., 2011). The strings presented to our grammatical learner contain a prefix which encodes the objects and their social cues for each utterance, and the rules of the grammar encode relationships between these objects and specific words. These rules permit every object to map to every word (including function words; i.e., there is no “stop word” list), and the learning process decides which of these rules will have a non-trivial probability (these encode the object-word mappings the system has learned).

This reduction of grounded learning to grammatical inference allows us to use standard grammatical inference procedures to learn our models. Here we use the *adaptor grammar* package described in Johnson et al. (2007) and Johnson and Goldwater (2009) with “out of the box” default settings; no parameter tuning whatsoever was done. Adaptor grammars are a framework for specifying hierarchical non-parametric models that has been previously used to model language acquisition (Johnson, 2008).

Social cue	Value
<i>child.eyes</i>	objects child is looking at
<i>child.hands</i>	objects child is touching
<i>mom.eyes</i>	objects care-giver is looking at
<i>mom.hands</i>	objects care-giver is touching
<i>mom.point</i>	objects care-giver is pointing to

Figure 1: The 5 social cues in the Frank et al. (to appear) corpus. The value of a social cue for an utterance is a subset of the available topics (i.e., the objects in the non-linguistic context) of that utterance.

A semanticist might argue that our view of referential mapping is flawed: full noun phrases (e.g., *the dog*), rather than nouns, refer to specific objects, and nouns denote properties (e.g., *dog* denotes the property of being a dog). Learning that a noun, e.g., *dog*, is part of a phrase used to refer to a specific dog (say, Fido) does not suffice to determine the noun’s meaning: the noun could denote a specific breed of dog, or animals in general. But learning word-object relationships is a plausible first step for any learner: it is often only the contrast between learned relationships and novel relationships that allows children to induce super- or sub-ordinate mappings (Clark, 1987). Nevertheless, in deference to such objections, we call the object that a phrase containing a given noun refers to the *topic* of that noun. (This is also appropriate, given that our models are specialisations of topic models).

Our models are intended as an “ideal learner” approach to early social language learning, attempting to weight the importance of social and structural factors in the acquisition of word-object correspondences. From this perspective, the primary goal is to investigate the relationships between acquisition tasks (Johnson, 2008; Johnson et al., 2010), looking for synergies (areas of acquisition where attempting two learning tasks jointly can provide gains in both) as well as areas where information overlaps.

### 1.1 A training corpus for social cues

Our work here uses a corpus of child-directed speech annotated with social cues, described in Frank et al. (to appear). The corpus consists of 4,763 orthographically-transcribed utterances of caregivers to their pre-linguistic children (ages 6, 12, and 18 months) during home visits where children played with a consistent set of toys. The sessions were video-taped, and each utterance was annotated with the five social cues described in Figure 1.

Each utterance in the corpus contains the follow-

ing information:

- the sequence of orthographic words uttered by the care-giver,
- a set of *available topics* (i.e., objects in the non-linguistic objects),
- the values of the social cues, and
- a set of *intended topics*, which the care-giver refers to.

Figure 2 presents this information for an example utterance. All of these but the intended topics are provided to our learning algorithms; the intended topics are used to evaluate the output produced by our learners.

Generally the intended topics consist of zero or one elements from the available topics, but not always: it is possible for the caregiver to refer to two objects in a single utterance, or to refer to an object not in the current non-linguistic context (e.g., to a toy that has been put away). There is a considerable amount of anaphora in this corpus, which our models currently ignore.

Frank et al. (to appear) give extensive details on the corpus, including inter-annotator reliability information for all annotations, and provide detailed statistical analyses of the relationships between the various social cues, the available topics and the intended topics. That paper also gives instructions on obtaining the corpus.

### 1.2 Previous work

There is a growing body of work on the role of social cues in language acquisition. The language acquisition research community has long recognized the importance of social cues for child language acquisition (Baldwin, 1991; Carpenter et al., 1998; Kuhl et al., 2003).

Siskind (1996) describes one of the first examples of a model that learns the relationship between words and topics, albeit in a non-statistical framework. Yu and Ballard (2007) describe an associative learner that associates words with topics and that exploits prosodic as well as social cues. The relative importance of the various social cues are specified a priori in their model (rather than learned, as they are here), and unfortunately their training corpus is not available. Frank et al. (2008) describes a Bayesian model that learns the relationship between words and topics, but the version of their model that included social cues presented a number of challenges for inference. The unigram model we describe below corresponds most closely to the Frank





*.dog # .pig child.eyes mom.eyes mom.hands # ## wheres the piggie*

Figure 2: The photograph indicates non-linguistic context containing a (toy) pig and dog for the utterance *Where's the piggie?*. Below that, we show the representation of this utterance that serves as the input to our models. The prefix (the portion of the string before the “##”) lists the available topics (i.e., the objects in the non-linguistic context) and their associated social cues (the cues for the pig are *child.eyes*, *mom.eyes* and *mom.hands*, while the dog is not associated with any social cues). The intended topic is the pig. The learner's goals are to identify the utterance's intended topic, and which words in the utterance are associated with which topic.

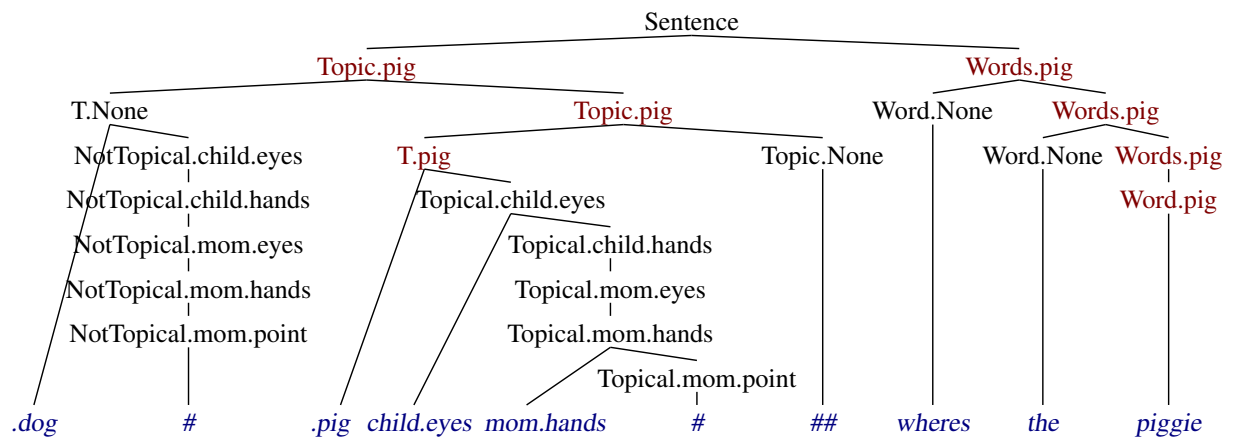


Figure 3: Sample parse generated by the Unigram PCFG. Nodes coloured red show how the “pig” topic is propagated from the prefix (before the “##” separator) into the utterance. The social cues associated with each object are generated either from a “Topical” or a “NotTopical” nonterminal, depending on whether the corresponding object is topical or not.

et al. model. Johnson et al. (2010) reduces grounded learning to grammatical inference for adaptor grammars and shows how it can be used to perform word segmentation as well as learning word-topic relationships, but their model does not take social cues into account.

## 2 Reducing grounded learning with social cues to grammatical inference

This section explains how we reduce ground learning problems with social cues to grammatical inference problems, which lets us apply a wide variety of grammatical inference algorithms to grounded learning problems. An advantage of reducing grounded learning to grammatical inference is that it suggests new ways to generalise grounded learning models; we explore three such generalisations here. The main challenge in this reduction is finding a way of expressing the non-linguistic information as part of the strings that serve as the grammatical inference procedure’s input. Here we encode the non-linguistic information in a “prefix” to each utterance as shown in Figure 2, and devise a grammar such that inference for the grammar corresponds to learning the word-topic relationships and the salience of the social cues for grounded learning.

All our models associate each utterance with zero or one topics (this means we cannot correctly analyse utterances with more than one intended topic). We analyse an utterance associated with zero topics as having the special topic None, so we can assume that every utterance has exactly one topic. All our grammars generate strings of the form shown in Figure 2, and they do so by parsing the prefix and the words of the utterance separately; the top-level rules of the grammar force the same topic to be associated with both the prefix and the words of the utterance (see Figure 3).

### 2.1 Topic models and the unigram PCFG

As Johnson et al. (2010) observe, this kind of grounded learning can be viewed as a specialised kind of topic inference in a topic model, where the utterance topic is constrained by the available objects (possible topics). We exploit this observation here using a reduction based on the reduction of LDA topic models to PCFGs proposed by Johnson (2010). This leads to our first model, the unigram grammar, which is a PCFG.<sup>1</sup>

<sup>1</sup>In fact, the unigram grammar is equivalent to a HMM, but the PCFG parameterisation makes clear the relationship

Sentence	$\rightarrow$ Topic <sub><i>t</i></sub> Words <sub><i>t</i></sub>	$\forall t \in T'$
Topic <sub>None</sub>	$\rightarrow$ ##	
Topic <sub><i>t</i></sub>	$\rightarrow$ T <sub><i>t</i></sub> Topic <sub>None</sub>	$\forall t \in T'$
Topic <sub><i>t</i></sub>	$\rightarrow$ T <sub>None</sub> Topic <sub><i>t</i></sub>	$\forall t \in T$
T <sub><i>t</i></sub>	$\rightarrow$ <i>t</i> Topical <sub><i>c</i><sub>1</sub></sub>	$\forall t \in T$
Topical <sub><i>c</i><sub><i>i</i></sub></sub>	$\rightarrow$ ( <i>c</i> <sub><i>i</i></sub> ) Topical <sub><i>c</i><sub><i>i</i>+1</sub></sub>	$i = 1, \dots, \ell - 1$
Topical <sub><i>c</i><sub><math>\ell</math></sub></sub>	$\rightarrow$ ( <i>c</i> <sub><math>\ell</math></sub> ) #	
T <sub>None</sub>	$\rightarrow$ <i>t</i> NotTopical <sub><i>c</i><sub>1</sub></sub>	$\forall t \in T$
NotTopical <sub><i>c</i><sub><i>i</i></sub></sub>	$\rightarrow$ ( <i>c</i> <sub><i>i</i></sub> ) NotTopical <sub><i>c</i><sub><i>i</i>+1</sub></sub>	$i = 1, \dots, \ell - 1$
NotTopical <sub><i>c</i><sub><math>\ell</math></sub></sub>	$\rightarrow$ ( <i>c</i> <sub><math>\ell</math></sub> ) #	
Words <sub><i>t</i></sub>	$\rightarrow$ Word <sub>None</sub> (Words <sub><i>t</i></sub> )	$\forall t \in T'$
Words <sub><i>t</i></sub>	$\rightarrow$ Word <sub><i>t</i></sub> (Words <sub><i>t</i></sub> )	$\forall t \in T$
Word <sub><i>t</i></sub>	$\rightarrow$ <i>w</i>	$\forall t \in T', w \in W$

Figure 4: The rule schema that generate the unigram PCFG. Here (*c*<sub>1</sub>, . . . , *c* <sub>$\ell$</sub> ) is an ordered list of the social cues, *T* is the set of all non-None available topics,  $T' = T \cup \{\text{None}\}$ , and *W* is the set of words appearing in the utterances. Parentheses indicate optionality.

Figure 4 presents the rules of the unigram grammar. This grammar has two major parts. The rules expanding the Topic<sub>*t*</sub> nonterminals ensure that the social cues for the available topic *t* are parsed under the Topical nonterminals. All other available topics are parsed under T<sub>None</sub> nonterminals, so their social cues are parsed under NotTopical nonterminals. The rules expanding these non-terminals are specifically designed so that the generation of the social cues corresponds to a series of binary decisions about each social cue. For example, the probability of the rule

$$\text{Topical}_{child.eyes} \rightarrow .child.eyes \text{Topical}_{child.hands}$$

is the probability of an object that is an utterance topic occurring with the *child.eyes* social cue. By estimating the probabilities of these rules, the model effectively learns the probability of each social cue being associated with a Topical or a NotTopical available topic, respectively.

The nonterminals Words<sub>*t*</sub> expand to a sequence of Word<sub>*t*</sub> and Word<sub>None</sub> nonterminals, each of which can expand to any word whatsoever. In practice Word<sub>*t*</sub> will expand to those words most strongly associated with topic *t*, while Word<sub>None</sub> will expand to those words not associated with any topic.

between grounded learning and estimation of grammar rule weights.

Sentence	$\rightarrow$ Topic <sub>t</sub> Colloc <sub>t</sub>	$\forall t \in T'$
Colloc <sub>t</sub>	$\rightarrow$ Colloc <sub>t</sub> (Colloc <sub>s</sub> <sub>t</sub> )	$\forall t \in T'$
Colloc <sub>s</sub> <sub>t</sub>	$\rightarrow$ Colloc <sub>None</sub> (Colloc <sub>s</sub> <sub>t</sub> )	$\forall t \in T$
<u>Colloc<sub>t</sub></u>	$\rightarrow$ Words <sub>t</sub>	$\forall t \in T'$
<u>Words<sub>t</sub></u>	$\rightarrow$ Word <sub>t</sub> (Words <sub>t</sub> )	$\forall t \in T'$
Words <sub>s</sub> <sub>t</sub>	$\rightarrow$ Word <sub>None</sub> (Words <sub>s</sub> <sub>t</sub> )	$\forall t \in T$
<u>Word<sub>t</sub></u>	$\rightarrow$ Word	$\forall t \in T'$
Word	$\rightarrow$ w	$\forall w \in W$

Figure 5: The rule schema that generate the collocation adaptor grammar. Adapted nonterminals are indicated via underlining. Here  $T$  is the set of all non-None available topics,  $T' = T \cup \{\text{None}\}$ , and  $W$  is the set of words appearing in the utterances. The rules expanding the Topic<sub>t</sub> nonterminals are exactly as in unigram PCFG.

## 2.2 Adaptor grammars

Our other grounded learning models are based on reductions of grounded learning to adaptor grammar inference problems. Adaptor grammars are a framework for stating a variety of Bayesian non-parametric models defined in terms of a hierarchy of Pitman-Yor Processes: see Johnson et al. (2007) for a formal description. Informally, an adaptor grammar is specified by a set of rules just as in a PCFG, plus a set of *adapted nonterminals*. The set of trees generated by an adaptor grammar is the same as the set of trees generated by a PCFG with the same rules, but the generative process differs. Non-adapted nonterminals in an adaptor grammar expand just as they do in a PCFG: the probability of choosing a rule is specified by its probability. However, the expansion of an adapted nonterminal depends on how it expanded in previous derivations. An adapted nonterminal can directly expand to a subtree with probability proportional to the number of times that subtree has been previously generated; it can also “back off” to expand using a grammar rule, just as in a PCFG, with probability proportional to a constant.<sup>2</sup>

Thus an adaptor grammar can be viewed as caching each tree generated by each adapted nonterminal, and regenerating it with probability proportional to the number of times it was previously generated (with some probability mass reserved to generate “new” trees). This enables adaptor gram-

<sup>2</sup>This is a description of Chinese Restaurant Processes, which are the predictive distributions for Dirichlet Processes. Our adaptor grammars are actually based on the more general Pitman-Yor Processes, as described in Johnson and Goldwater (2009).

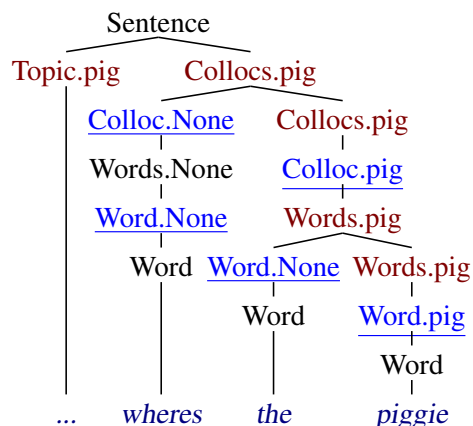


Figure 6: Sample parse generated by the collocation adaptor grammar. The adapted nonterminals Colloc<sub>t</sub> and Word<sub>t</sub> are shown underlined; the subtrees they dominate are “cached” by the adaptor grammar. The prefix (not shown here) is parsed exactly as in the Unigram PCFG.

mars to generalise over subtrees of arbitrary size. Generic software is available for adaptor grammar inference, based either on Variational Bayes (Cohen et al., 2010) or Markov Chain Monte Carlo (Johnson and Goldwater, 2009). We used the latter software because it is capable of performing hyper-parameter inference for the PCFG rule probabilities and the Pitman-Yor Process parameters. We used the “out-of-the-box” settings for this software, i.e., uniform priors on all PCFG rule parameters, a Beta(2, 1) prior on the Pitman-Yor  $a$  parameters and a “vague” Gamma(100, 0.01) prior on the Pitman-Yor  $b$  parameters. (Presumably performance could be improved if the priors were tuned, but we did not explore this here).

Here we explore a simple “collocation” extension to the unigram PCFG which associates multiword collocations, rather than individual words, with topics. Hardisty et al. (2010) showed that this significantly improved performance in a sentiment analysis task.

The collocation adaptor grammar in Figure 5 generates the words of the utterance as a sequence of collocations, each of which is a sequence of words. Each collocation is either associated with the sentence topic or with the None topic, just like words in the unigram model. Figure 6 shows a sample parse generated by the collocation adaptor grammar.

We also experimented with a variant of the unigram and collocation grammars in which the topic-specific word distributions Word<sub>t</sub> for each  $t \in T$

Model	Social cues	Utterance topic				Word topic			Lexicon		
		acc.	f-score	prec.	rec.	f-score	prec.	rec.	f-score	prec.	rec.
unigram	none	0.3395	0.4044	0.3249	0.5353	0.2007	0.1207	0.5956	0.1037	0.05682	0.5952
unigram	all	0.4907	0.6064	0.4867	<b>0.8043</b>	0.295	0.1763	<b>0.9031</b>	0.1483	0.08096	<b>0.881</b>
colloc	none	0.4331	0.3513	0.3272	0.3792	0.2431	0.1603	0.5028	0.08808	0.04942	0.4048
colloc	all	0.5837	0.598	0.5623	0.6384	<b>0.4098</b>	<b>0.2702</b>	0.8475	0.1671	0.09422	0.7381
unigram'	none	0.3261	0.3767	0.3054	0.4914	0.1893	0.1131	0.5811	0.1167	0.06583	0.5122
unigram'	all	0.5117	<b>0.6106</b>	0.4986	0.7875	0.2846	0.1693	0.891	0.1684	0.09402	0.8049
colloc'	none	0.5238	0.3419	0.3844	0.3078	0.2551	0.1732	0.4843	0.2162	0.1495	0.3902
colloc'	all	<b>0.6492</b>	0.6034	<b>0.6664</b>	0.5514	0.3981	0.2613	0.8354	<b>0.3375</b>	<b>0.2269</b>	0.6585

Figure 7: Utterance topic, word topic and lexicon results for all models, on data with and without social cues. The results for the variant models, in which  $\text{Word}_t$  nonterminals expand via  $\text{Word}_{\text{None}}$ , are shown under unigram' and colloc'. Utterance topic shows how well the model discovered the intended topics at the utterance level, word topic shows how well the model associates word tokens with topics, and lexicon shows how well the topic most frequently associated with a word type matches an external word-topic dictionary. In this figure and below, “colloc” abbreviates “collocation”, “acc.” abbreviates “accuracy”, “prec.” abbreviates “precision” and “rec.” abbreviates “recall”.

(the set of non-None available topics) expand via  $\text{Word}_{\text{None}}$  non-terminals. That is, in the variant grammars topical words are generated with the following rule schema:

$$\begin{aligned} \text{Word}_t &\rightarrow \text{Word}_{\text{None}} && \forall t \in T \\ \text{Word}_{\text{None}} &\rightarrow \text{Word} \\ \text{Word} &\rightarrow w && \forall w \in W \end{aligned}$$

In these variant grammars, the  $\text{Word}_{\text{None}}$  nonterminal generates all the words of the language, so it defines a generic “background” distribution over all the words, rather than just the nontopical words. An effect of this is that the variant grammars tend to identify fewer words as topical.

### 3 Experimental evaluation

We performed grammatical inference using the adaptor grammar software described in Johnson and Goldwater (2009).<sup>3</sup> All experiments involved 4 runs of 5,000 samples each, of which the first 2,500 were discarded for “burn-in”.<sup>4</sup> From these samples we extracted the modal (i.e., most frequent) analysis,

<sup>3</sup>Because adaptor grammars are a generalisation of PCFGs, we could use the adaptor grammar software to estimate the unigram model.

<sup>4</sup>We made no effort to optimise the computation, but it seems the samplers actually stabilised after around a hundred iterations, so it was probably not necessary to sample so extensively. We estimated the error in our results by running our most complex model (the colloc' model with all social cues) 20 times (i.e.,  $20 \times 8$  chains for 5,000 iterations) so we could compute the variance of each of the evaluation scores (it is reasonable to assume that the simpler models will have smaller variance). The standard deviation of all utterance topic and word topic measures is between 0.005 and 0.01; the standard deviation for lexicon f-score is 0.02, lexicon precision is 0.01 and lexicon recall is 0.03. The adaptor grammar software uses a sentence-wise

which we evaluated as described below. The results of evaluating each model on the corpus with social cues, and on another corpus identical except that the social cues have been removed, are presented in Figure 7.

Each model was evaluated on each corpus as follows. First, we extracted the utterance’s topic from the modal parse (this can be read off the  $\text{Topic}_t$  nodes), and compared this to the intended topics annotated in the corpus. The frequency with which the models’ predicted topics exactly matches the intended topics is given under “utterance topic accuracy”; the f-score, precision and recall of each model’s topic predictions are also given in the table.

Because our models all associate word tokens with topics, we can also evaluate the accuracy with which word tokens are associated with topics. We constructed a small dictionary which identifies the words that can be used as the head of a phrase to refer to the topical objects (e.g., the dictionary indicates that *dog*, *doggie* and *puppy* name the topical object DOG). Our dictionary is relatively conservative; between one and eight words are associated with each topic. We scored the topic label on each word token in our corpus as follows. A topic label is scored as correct if it is given in our dictionary and the topic is one of the intended topics for the utterance. The “word topic” entries in Figure 7 give the results of this evaluation.

blocked sampler, so it requires fewer iterations than a pointwise sampler. We used 5,000 iterations because this is the software’s default setting; evaluating the trace output suggests it only takes several hundred iterations to “burn in”. However, we ran 8 chains for 25,000 iterations of the colloc' model; as expected the results of this run are within two standard deviations of the results reported above.

Model	Social cues	Utterance topic				Word topic			Lexicon		
		acc.	f-score	prec.	rec.	f-score	prec.	rec.	f-score	prec.	rec.
unigram	none	0.3395	0.4044	0.3249	0.5353	0.2007	0.1207	0.5956	0.1037	0.05682	0.5952
unigram	+ <i>child.eyes</i>	<b>0.4573</b>	<b>0.5725</b>	<b>0.4559</b>	<b>0.7694</b>	<b>0.2891</b>	<b>0.1724</b>	<b>0.8951</b>	<b>0.1362</b>	<b>0.07415</b>	<b>0.8333</b>
unigram	+ <i>child.hands</i>	0.3399	0.4011	0.3246	0.5247	0.2008	0.121	0.5892	0.09705	0.05324	0.5476
unigram	+ <i>mom.eyes</i>	0.338	0.4023	0.3234	0.5322	0.1992	0.1198	0.5908	0.09664	0.053	0.5476
unigram	+ <i>mom.hands</i>	0.3563	0.4279	0.3437	0.5667	0.1984	0.1191	0.5948	0.09959	0.05455	0.5714
unigram	+ <i>mom.point</i>	0.3063	0.3548	0.285	0.4698	0.1806	0.1086	0.5359	0.09224	0.05057	0.5238
colloc	none	0.4331	0.3513	0.3272	0.3792	0.2431	0.1603	0.5028	0.08808	0.04942	0.4048
colloc	+ <i>child.eyes</i>	<b>0.5159</b>	<b>0.5006</b>	<b>0.4652</b>	<b>0.542</b>	<b>0.351</b>	<b>0.2309</b>	<b>0.7312</b>	<b>0.1432</b>	<b>0.07989</b>	<b>0.6905</b>
colloc	+ <i>child.hands</i>	0.4827	0.4275	0.3999	0.4592	0.2897	0.1913	0.5964	0.1192	0.06686	0.5476
colloc	+ <i>mom.eyes</i>	0.4697	0.4171	0.3869	0.4525	0.2708	0.1781	0.5642	0.1013	0.05666	0.4762
colloc	+ <i>mom.hands</i>	0.4747	0.4251	0.3942	0.4612	0.274	0.1806	0.5666	0.09548	0.05337	0.4524
colloc	+ <i>mom.point</i>	0.4228	0.3378	0.3151	0.3639	0.2575	0.1716	0.5157	0.09278	0.05202	0.4286

Figure 8: Effect of using just one social cue on the experimental results for the unigram and collocation models. The “importance” of a social cue can be quantified by the degree to which the model’s evaluation score improves when using a corpus containing that social cue relative to its evaluation score when using a corpus without any social cues. The most important social cue is the one which causes performance to improve the most.

Finally, we extracted a lexicon from the parsed corpus produced by each model. We counted how often each word type was associated with each topic in our sampler’s output (including the None topic), and assigned the word to its most frequent topic. The “lexicon” entries in Figure 7 show how well the entries in these lexicons match the entries in the manually-constructed dictionary discussed above.

There are 10 different evaluation scores, and no model dominates in all of them. However, the top-scoring result in every evaluation is always for a model trained using social cues, demonstrating the importance of these social cues. The variant collocation model (trained on data with social cues) was the top-scoring model on four evaluation scores, which is more than any other model.

One striking thing about this evaluation is that the recall scores are all much higher than the precision scores, for each evaluation. This indicates that all of the models, especially the unigram model, are labelling too many words as topical. This is perhaps not too surprising: because our models completely lack any notion of syntactic structure and simply model the association between words and topics, they label many non-nouns with topics (e.g., *woof* is typically labelled with the topic DOG).

### 3.1 Evaluating the importance of social cues

It is scientifically interesting to be able to evaluate the importance of each of the social cues to grounded learning. One way to do this is to study the effect of adding or removing social cues from the corpus on the ability of our models to perform grounded learning. An important social cue should

have a large impact on our models’ performance; an unimportant cue should have little or no impact.

Figure 8 compares the performance of the unigram and collocation models on corpora containing a single social cue to their performance on the corpus without any social cues, while Figure 9 compares the performance of these models on corpora containing all but one social cue to the corpus containing all of the social cues. In both of these evaluations, with respect to all 10 evaluation measures, the *child.eyes* social cue had the most impact on model performance.

Why would the child’s own gaze be more important than the caregiver’s? Perhaps caregivers are *following in*, i.e., talking about objects that their children are interested in (Baldwin, 1991). However, another possible explanation is that this result is due to the general continuity of conversational topics over time. Frank et al. (to appear) show that for the current corpus, the topic of the preceding utterance is very likely to be the topic of the current one also. Thus, the child’s eyes might be a good predictor because they reflect the fact that the child’s attention has been drawn to an object by previous utterances.

Notice that these two possible explanations of the importance of the *child.eyes* cue are diametrically opposed; the first explanation claims that the cue is important because the child is driving the discourse, while the second explanation claims that the cue is important because the child’s gaze follows the topic of the caregiver’s previous utterance. This sort of question about causal relationships in conversations may be very difficult to answer using standard descriptive techniques, but it may be an interesting av-

Model	Social cues	Utterance topic				Word topic			Lexicon		
		acc.	f-score	prec.	rec.	f-score	prec.	rec.	f-score	prec.	rec.
unigram	all	0.4907	0.6064	0.4867	0.8043	0.295	0.1763	0.9031	0.1483	0.08096	0.881
unigram	– <i>child.eyes</i>	<b>0.3836</b>	<b>0.4659</b>	<b>0.3738</b>	<b>0.6184</b>	<b>0.2149</b>	<b>0.1286</b>	<b>0.6546</b>	<b>0.1111</b>	<b>0.06089</b>	<b>0.6341</b>
unigram	– <i>child.hands</i>	0.4907	0.6063	0.4863	0.8051	0.296	0.1769	0.9056	0.1525	0.08353	0.878
unigram	– <i>mom.eyes</i>	0.4799	0.5974	0.4768	0.7996	0.2898	0.1727	0.9007	0.1551	0.08486	0.9024
unigram	– <i>mom.hands</i>	0.4871	0.5996	0.4815	0.7945	0.2925	0.1746	0.8991	0.1561	0.08545	0.9024
unigram	– <i>mom.point</i>	0.4875	0.6033	0.4841	0.8004	0.2934	0.1752	0.9007	0.1558	0.08525	0.9024
colloc	all	0.5837	0.598	0.5623	0.6384	0.4098	0.2702	0.8475	0.1671	0.09422	0.738
colloc	– <i>child.eyes</i>	<b>0.5604</b>	<b>0.5746</b>	<b>0.529</b>	<b>0.6286</b>	<b>0.39</b>	<b>0.2561</b>	<b>0.8176</b>	<b>0.1534</b>	<b>0.08642</b>	<b>0.6829</b>
colloc	– <i>child.hands</i>	0.5849	0.6	0.5609	0.6451	0.4145	0.273	0.8612	0.1662	0.09375	0.7317
colloc	– <i>mom.eyes</i>	0.5709	0.5829	0.5457	0.6255	0.4036	0.2655	0.8418	0.1662	0.09375	0.7317
colloc	– <i>mom.hands</i>	0.5795	0.5935	0.5571	0.6349	0.4038	0.2653	0.8442	0.1788	0.1009	0.7805
colloc	– <i>mom.point</i>	0.5851	0.6006	0.5607	0.6467	0.4097	0.2685	0.8644	0.1742	0.09841	0.7561

Figure 9: Effect of using all but one social cue on the experimental results for the unigram and collocation models. The “importance” of a social cue can be quantified by the degree to which the model’s evaluation score degrades when that just social cue is removed from the corpus, relative to its evaluation score when using a corpus without all social cues. The most important social cue is the one which causes performance to degrade the most.

enue for future investigation using more structured models such as those proposed here.<sup>5</sup>

#### 4 Conclusion and future work

This paper presented four different grounded learning models that exploit social cues. These models are all expressed via reductions to grammatical inference problems, so standard “off the shelf” grammatical inference tools can be used to learn them. Here we used the same adaptor grammar software tools to learn all these models, so we can be relatively certain that any differences we observe are due to differences in the models, rather than quirks in the software.

Because the adaptor grammar software performs full Bayesian inference, including for model parameters, an unusual feature of our models is that we did not need to perform any parameter tuning whatsoever. This feature is particularly interesting with respect to the parameters on social cues. Psychological proposals have suggested that children may discover that particular social cues help in establishing reference (Baldwin, 1993; Hollich et al., 2000), but prior modeling work has often assumed that cues, cue weights, or both are prespecified. In contrast, the models described here could in principle discover a wide range of different social conventions.

<sup>5</sup>A reviewer suggested that we can test whether *child.eyes* effectively provides the same information as the previous topic by adding the previous topic as a (pseudo-) social cue. We tried this, and *child.eyes* and *previous.topic* do in fact seem to convey very similar information: e.g., the model with *previous.topic* and without *child.eyes* scores essentially the same as the model with all social cues.

Our work instantiates the strategy of investigating the structure of children’s learning environment using “ideal learner” models. We used our models to investigate scientific questions about the role of social cues in grounded language learning. Because the performance of all four models studied in this paper improve dramatically when provided with social cues in all ten evaluation metrics, this paper provides strong support for the view that social cues are a crucial information source for grounded language learning.

We also showed that the importance of the different social cues in grounded language learning can be evaluated using “add one cue” and “subtract one cue” methodologies. According to both of these, the *child.eyes* cue is the most important of the five social cues studied here. There are at least two possible reasons for this: the caregiver’s topic could be determined by the child’s gaze, or the *child.eyes* cue could be providing our models with information about the topic of the previous utterance.

Incorporating topic continuity and anaphoric dependencies into our models would be likely to improve performance. This improvement might also help us distinguish the two hypotheses about the *child.eyes* cue. If the *child.eyes* cue is just providing indirect information about topic continuity, then the importance of the *child.eyes* cue should decrease when we incorporate topic continuity into our models. But if the child’s gaze is in fact determining the care-giver’s topic, then *child.eyes* should remain a strong cue even when anaphoric dependencies and topic continuity are incorporated into our models.

## Acknowledgements

This research was supported under the Australian Research Council's *Discovery Projects* funding scheme (project number DP110102506).

## References

- Dare A. Baldwin. 1991. Infants' contribution to the achievement of joint reference. *Child Development*, 62(5):874–890.
- Dare A. Baldwin. 1993. Infants' ability to consult the speaker for clues to word reference. *Journal of Child Language*, 20:395–395.
- Benjamin Börschinger, Bevan K. Jones, and Mark Johnson. 2011. Reducing grounded learning tasks to grammatical inference. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1416–1425, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- M. Carpenter, K. Nagell, M. Tomasello, G. Butterworth, and C. Moore. 1998. Social cognition, joint attention, and communicative competence from 9 to 15 months of age. *Monographs of the society for research in child development*.
- E.V. Clark. 1987. The principle of contrast: A constraint on language acquisition. *Mechanisms of language acquisition*, 1:33.
- Shay B. Cohen, David M. Blei, and Noah A. Smith. 2010. Variational inference for adaptor grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 564–572, Los Angeles, California, June. Association for Computational Linguistics.
- Michael Frank, Noah Goodman, and Joshua Tenenbaum. 2008. A Bayesian framework for cross-situational word-learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 457–464, Cambridge, MA. MIT Press.
- Michael C. Frank, Joshua Tenenbaum, and Anne Fernald. to appear. Social and discourse contributions to the determination of reference in cross-situational word learning. *Language, Learning, and Development*.
- Eric A. Hardisty, Jordan Boyd-Graber, and Philip Resnik. 2010. Modeling perspective using adaptor grammars. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 284–292, Stroudsburg, PA, USA. Association for Computational Linguistics.
- G.J. Hollich, K. Hirsh-Pasek, and R. Golinkoff. 2000. Breaking the language barrier: An emergentist coalition model for the origins of word learning. *Monographs of the Society for Research in Child Development*.
- Mark Johnson and Sharon Goldwater. 2009. Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, Boulder, Colorado, June. Association for Computational Linguistics.
- Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. Adaptor Grammars: A framework for specifying compositional nonparametric Bayesian models. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, Cambridge, MA.
- Mark Johnson, Katherine Demuth, Michael Frank, and Bevan Jones. 2010. Synergies in learning words and their referents. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1018–1026.
- Mark Johnson. 2008. Using adaptor grammars to identifying synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, pages 398–406, Columbus, Ohio. Association for Computational Linguistics.
- Mark Johnson. 2010. PCFGs, topic models, adaptor grammars and learning topical collocations and the structure of proper names. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1148–1157, Uppsala, Sweden, July. Association for Computational Linguistics.
- Patricia K. Kuhl, Feng-Ming Tsao, and Huei-Mei Liu. 2003. Foreign-language experience in infancy: Effects of short-term exposure and social interaction on phonetic learning. *Proceedings of the National Academy of Sciences USA*, 100(15):9096–9101.
- Jeffrey Siskind. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, 61(1-2):39–91.
- L.B. Smith, S.S. Jones, B. Landau, L. Gershkoff-Stowe, and L. Samuelson. 2002. Object name learning provides on-the-job training for attention. *Psychological Science*, 13(1):13.
- Chen Yu and Dana H Ballard. 2007. A unified model of early word learning: Integrating statistical and social cues. *Neurocomputing*, 70(13-15):2149–2165.



# You Had Me at Hello: How Phrasing Affects Memorability

Cristian Danescu-Niculescu-Mizil Justin Cheng Jon Kleinberg Lillian Lee

Department of Computer Science

Cornell University

cristian@cs.cornell.edu, jc882@cornell.edu, kleinber@cs.cornell.edu, llee@cs.cornell.edu

## Abstract

Understanding the ways in which information achieves widespread public awareness is a research question of significant interest. We consider whether, and how, the way in which the information is phrased — the choice of words and sentence structure — can affect this process. To this end, we develop an analysis framework and build a corpus of movie quotes, annotated with memorability information, in which we are able to control for both the speaker and the setting of the quotes. We find that there are significant differences between memorable and non-memorable quotes in several key dimensions, even after controlling for situational and contextual factors. One is *lexical distinctiveness*: in aggregate, memorable quotes use less common word choices, but at the same time are built upon a scaffolding of common syntactic patterns. Another is that memorable quotes tend to be more *general* in ways that make them easy to apply in new contexts — that is, more portable. We also show how the concept of “memorable language” can be extended across domains.

## 1 Hello. My name is Inigo Montoya.

Understanding what items will be retained in the public consciousness, and why, is a question of fundamental interest in many domains, including marketing, politics, entertainment, and social media; as we all know, many items barely register, whereas others catch on and take hold in many people’s minds.

An active line of recent computational work has employed a variety of perspectives on this question.

Building on a foundation in the sociology of diffusion [27, 31], researchers have explored the ways in which network structure affects the way information spreads, with domains of interest including blogs [1, 11], email [37], on-line commerce [22], and social media [2, 28, 33, 38]. There has also been recent research addressing temporal aspects of how different media sources convey information [23, 30, 39] and ways in which people react differently to information on different topics [28, 36].

Beyond all these factors, however, one’s everyday experience with these domains suggests that the way in which a piece of information is expressed — the choice of words, the way it is phrased — might also have a fundamental effect on the extent to which it takes hold in people’s minds. Concepts that attain wide reach are often carried in messages such as political slogans, marketing phrases, or aphorisms whose language seems intuitively to be memorable, “catchy,” or otherwise compelling.

Our first challenge in exploring this hypothesis is to develop a notion of “successful” language that is precise enough to allow for quantitative evaluation. We also face the challenge of devising an evaluation setting that separates the phrasing of a message from the conditions in which it was delivered — highly-cited quotes tend to have been delivered under compelling circumstances or fit an existing cultural, political, or social narrative, and potentially what appeals to us about the quote is really just its invocation of these extra-linguistic contexts. Is the form of the language adding an effect *beyond or independent of* these (obviously very crucial) factors? To investigate the question, one needs a way of control-



ling — as much as possible — for the role that the surrounding context of the language plays.

**The present work (i): Evaluating language-based memorability** Defining what makes an utterance memorable is subtle, and scholars in several domains have written about this question. There is a rough consensus that an appropriate definition involves elements of both *recognition* — people should be able to retain the quote and recognize it when they hear it invoked — and *production* — people should be motivated to refer to it in relevant situations [15]. One suggested reason for why some memes succeed is their ability to provoke emotions [16]. Alternatively, memorable quotes can be good for expressing the feelings, mood, or situation of an individual, a group, or a culture (the *zeitgeist*): “Certain quotes exquisitely capture the mood or feeling we wish to communicate to someone. We hear them ... and store them away for future use” [10].

None of these observations, however, serve as definitions, and indeed, we believe it desirable to not pre-commit to an abstract definition, but rather to adopt an operational formulation based on external human judgments. In designing our study, we focus on a domain in which (i) there is rich use of language, some of which has achieved deep cultural penetration; (ii) there already exist a large number of external human judgments — perhaps implicit, but in a form we can extract; and (iii) we can control for the setting in which the text was used.

Specifically, we use the complete scripts of roughly 1000 movies, representing diverse genres, eras, and levels of popularity, and consider which lines are the most “memorable”. To acquire memorability labels, for each sentence in each script, we determine whether it has been listed as a “memorable quote” by users of the widely-known IMDb (the Internet Movie Database), and also estimate the number of times it appears on the Web. Both of these serve as memorability metrics for our purposes.

When we evaluate properties of memorable quotes, we compare them with quotes that are not assessed as memorable, but were spoken by the same character, at approximately the same point in the same movie. This enables us to control in a fairly fine-grained way for the confounding effects of context discussed above: we can observe differences

that persist even after taking into account both the speaker and the setting.

In a pilot validation study, we find that human subjects are effective at recognizing the more IMDb-memorable of two quotes, even for movies they have not seen. This motivates a search for features intrinsic to the text of quotes that signal memorability. In fact, comments provided by the human subjects as part of the task suggested two basic forms that such textual signals could take: subjects felt that (i) memorable quotes often involve a *distinctive* turn of phrase; and (ii) memorable quotes tend to invoke *general* themes that aren’t tied to the specific setting they came from, and hence can be more easily invoked for future (out of context) uses. We test both of these principles in our analysis of the data.

**The present work (ii): What distinguishes memorable quotes** Under the controlled-comparison setting sketched above, we find that memorable quotes exhibit significant differences from non-memorable quotes in several fundamental respects, and these differences in the data reinforce the two main principles from the human pilot study. First, we show a concrete sense in which memorable quotes are indeed *distinctive*: with respect to lexical language models trained on the newswire portions of the Brown corpus [21], memorable quotes have significantly lower likelihood than their non-memorable counterparts. Interestingly, this distinctiveness takes place at the level of words, but not at the level of other syntactic features: the part-of-speech composition of memorable quotes is in fact more likely with respect to newswire. Thus, we can think of memorable quotes as consisting, in an aggregate sense, of unusual word choices built on a scaffolding of common part-of-speech patterns.

We also identify a number of ways in which memorable quotes convey greater *generality*. In their patterns of verb tenses, personal pronouns, and determiners, memorable quotes are structured so as to be more “free-standing,” containing fewer markers that indicate references to nearby text.

Memorable quotes differ in other interesting aspects as well, such as sound distributions.

Our analysis of memorable movie quotes suggests a framework by which the memorability of text in a range of different domains could be investigated.

We provide evidence that such cross-domain properties may hold, guided by one of our motivating applications in marketing. In particular, we analyze a corpus of advertising slogans, and we show that these slogans have significantly greater likelihood at both the word level and the part-of-speech level with respect to a language model trained on memorable movie quotes, compared to a corresponding language model trained on non-memorable movie quotes. This suggests that some of the principles underlying memorable text have the potential to apply across different areas.

**Roadmap** §2 lays the empirical foundations of our work: the design and creation of our movie-quotes dataset, which we make publicly available (§2.1), a pilot study with human subjects validating IMDb-based memorability labels (§2.2), and further study of incorporating search-engine counts (§2.3). §3 details our analysis and prediction experiments, using both movie-quotes data and, as an exploration of cross-domain applicability, slogans data. §4 surveys related work across a variety of fields. §5 briefly summarizes and indicates some future directions.

## 2 I’m ready for my close-up.

### 2.1 Data

To study the properties of memorable movie quotes, we need a source of movie lines and a designation of memorability. Following [8], we constructed a corpus consisting of all lines from roughly 1000 movies, varying in genre, era, and popularity; for each movie, we then extracted the list of quotes from IMDb’s *Memorable Quotes* page corresponding to the movie.<sup>1</sup>

A memorable quote in IMDb can appear either as an individual sentence spoken by one character, or as a multi-sentence line, or as a block of dialogue involving multiple characters. In the latter two cases, it can be hard to determine which particular portion is viewed as memorable (some involve a build-up to a punch line; others involve the follow-through after a well-phrased opening sentence), and so we focus in our comparisons on those memorable quotes that

<sup>1</sup>This extraction involved some edit-distance-based alignment, since the exact form of the line in the script can exhibit minor differences from the version typed into IMDb.

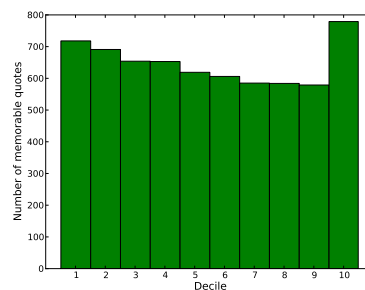


Figure 1: Location of memorable quotes in each decile of movie scripts (the first 10th, the second 10th, etc.), summed over all movies. The same qualitative results hold if we discard each movie’s very first and last line, which might have privileged status.

appear as a single sentence rather than a multi-line block.<sup>2</sup>

We now formulate a task that we can use to evaluate the features of memorable quotes. Recall that our goal is to identify effects based in the language of the quotes themselves, beyond any factors arising from the speaker or context. Thus, for each (single-sentence) memorable quote  $M$ , we identify a non-memorable quote that is as similar as possible to  $M$  in all characteristics but the choice of words. This means we want it to be spoken by the same character in the same movie. It also means that we want it to have the same length: controlling for length is important because we expect that on average, shorter quotes will be easier to remember than long quotes, and that wouldn’t be an interesting textual effect to report. Moreover, we also want to control for the fact that a quote’s position in a movie can affect memorability: certain scenes produce more memorable dialogue, and as Figure 1 demonstrates, in aggregate memorable quotes also occur disproportionately near the beginnings and especially the ends of movies. In summary, then, for each  $M$ , we pick a contrasting (single-sentence) quote  $N$  from the same movie that is as close in the script as possible to  $M$  (either before or after it), subject to the conditions that (i)  $M$  and  $N$  are uttered by the same speaker, (ii)  $M$  and  $N$  have the same number of words, and (iii)  $N$  does not occur in the IMDb list of memorable

<sup>2</sup>We also ran experiments relaxing the single-sentence assumption, which allows for stricter scene control and a larger dataset but complicates comparisons involving syntax. The non-syntax results were in line with those reported here.

Movie	First Quote	Second Quote
Jackie Brown	Half a million dollars will always be missed.	I know the type, trust me on this.
Star Trek: Nemesis	I think it's time to try some unsafe velocities.	No cold feet, or any other parts of our anatomy.
Ordinary People	A little advice about feelings kiddo; don't expect it always to tickle.	I mean there's someone besides your mother you've got to forgive.

Table 1: Three example pairs of movie quotes. Each pair satisfies our criteria: the two component quotes are spoken close together in the movie by the same character, have the same length, and one is labeled memorable by the IMDb while the other is not. (Contractions such as “it’s” count as two words.)

quotes for the movie (either as a single line or as part of a larger block).

Given such pairs, we formulate a *pairwise comparison task*: given  $M$  and  $N$ , determine which is the memorable quote. Psychological research on subjective evaluation [35], as well as initial experiments using ourselves as subjects, indicated that this pairwise set-up easier to work with than simply presenting a single sentence and asking whether it is memorable or not; the latter requires agreement on an “absolute” criterion for memorability that is very hard to impose consistently, whereas the former simply requires a judgment that one quote is more memorable than another.

Our main dataset, available at <http://www.cs.cornell.edu/~cristian/memorability.html>,<sup>3</sup> thus consists of approximately 2200 such  $(M, N)$  pairs, separated by a median of 5 same-character lines in the script. The reader can get a sense for the nature of the data from the three examples in Table 1.

We now discuss two further aspects to the formulation of the experiment: a preliminary pilot study involving human subjects, and the incorporation of search engine counts into the data.

## 2.2 Pilot study: Human performance

As a preliminary consideration, we did a small pilot study to see if humans can distinguish memorable from non-memorable quotes, assuming our IMDb-induced labels as gold standard. Six subjects, all native speakers of English and none an author of this paper, were presented with 11 or 12 pairs of memorable vs. non-memorable quotes; again, we controlled for extra-textual effects by ensuring that in each pair the two quotes come from the same movie, are by the same character, have the same length, and

<sup>3</sup>Also available there: other examples and factoids.

subject	number of matches with IMDb-induced annotation
A	11/11 = 100%
B	11/12 = 92%
C	9/11 = 82%
D	8/11 = 73%
E	7/11 = 64%
F	7/12 = 58%
macro avg	— 78%

Table 2: Human pilot study: number of matches to IMDb-induced annotation, ordered by decreasing match percentage. For the null hypothesis of random guessing, these results are statistically significant,  $p < 2^{-6} \approx .016$ .

appear as nearly as possible in the same scene.<sup>4</sup> The order of quotes within pairs was randomized. Importantly, because we wanted to understand whether the language of the quotes by itself contains signals about memorability, we chose quotes from movies that the subjects said they had not seen. (This means that each subject saw a different set of quotes.) Moreover, the subjects were requested not to consult any external sources of information.<sup>5</sup> The reader is welcome to try a demo version of the task at <http://www.cs.cornell.edu/~cristian/memorability.html>.

Table 2 shows that all the subjects performed (sometimes much) better than chance, and against the null hypothesis that all subjects are guessing randomly, the results are statistically significant,  $p < 2^{-6} \approx .016$ . These preliminary findings provide evidence for the validity of our task: despite the apparent difficulty of the job, even humans who haven’t seen the movie in question can recover our IMDb-

<sup>4</sup>In this pilot study, we allowed multi-sentence quotes.

<sup>5</sup>We did not use crowd-sourcing because we saw no way to ensure that this condition would be obeyed by arbitrary subjects. We do note, though, that after our research was completed and as of Apr. 26, 2012,  $\approx 11,300$  people completed the online test: average accuracy: 72%, mode number correct: 9/12.

induced labels with some reliability.<sup>6</sup>

### 2.3 Incorporating search engine counts

Thus far we have discussed a dataset in which memorability is determined through an explicit labeling drawn from the IMDb. Given the “production” aspect of memorability discussed in §1, we should also expect that memorable quotes will tend to appear more extensively on Web pages than non-memorable quotes; note that incorporating this insight makes it possible to use the (implicit) judgments of a much larger number of people than are represented by the IMDb database. It therefore makes sense to try using search-engine result counts as a second indication of memorability.

We experimented with several ways of constructing memorability information from search-engine counts, but this proved challenging. Searching for a quote as a stand-alone phrase runs into the problem that a number of quotes are also sentences that people use without the movie in mind, and so high counts for such quotes do not testify to the phrase’s status as a memorable quote from the movie. On the other hand, searching for the quote in a Boolean conjunction with the movie’s title discards most of these uses, but also eliminates a large fraction of the appearances on the Web that we want to find: precisely because memorable quotes tend to have widespread cultural usage, people generally don’t feel the need to include the movie’s title when invoking them. Finally, since we are dealing with roughly 1000 movies, the result counts vary over an enormous range, from recent blockbusters to movies with relatively small fan bases.

In the end, we found that it was more effective to use the result counts in conjunction with the IMDb labels, so that the counts played the role of an additional filter rather than a free-standing numerical value. Thus, for each pair  $(M, N)$  produced using the IMDb methodology above, we searched for each of  $M$  and  $N$  as quoted expressions in a Boolean conjunction with the title of the movie. We then kept only those pairs for which  $M$  (i) produced more than five results in our (quoted, conjoined) search, and (ii) produced at least twice as many results as the cor-

<sup>6</sup>The average accuracy being below 100% reinforces that context is very important, too.

responding search for  $N$ . We created a version of this filtered dataset using each of Google and Bing, and all the main findings were consistent with the results on the IMDb-only dataset. Thus, in what follows, we will focus on the main IMDb-only dataset, discussing the relationship to the dataset filtered by search engine counts where relevant (in which case we will refer to the +Google dataset).

## 3 Never send a human to do a machine’s job.

We now discuss experiments that investigate the hypotheses discussed in §1. In particular, we devise methods that can assess the distinctiveness and generality hypotheses and test whether there exists a notion of “memorable language” that operates across domains. In addition, we evaluate and compare the predictive power of these hypotheses.

### 3.1 Distinctiveness

One of the hypotheses we examine is whether the use of language in memorable quotes is to some extent unusual. In order to quantify the level of distinctiveness of a quote, we take a language-model approach: we model “common language” using the newswire sections of the Brown corpus [21]<sup>7</sup>, and evaluate how distinctive a quote is by evaluating its likelihood with respect to this model — the lower the likelihood, the more distinctive. In order to assess different levels of lexical and syntactic distinctiveness, we employ a total of six Laplace-smoothed<sup>8</sup> language models: 1-gram, 2-gram, and 3-gram word LMs and 1-gram, 2-gram and 3-gram part-of-speech<sup>9</sup> LMs.

We find strong evidence that from a lexical perspective, memorable quotes are more distinctive than their non-memorable counterparts. As indicated in Table 3, for each of our lexical “common language” models, in about 60% of the quote pairs, the memorable quote is more distinctive.

Interestingly, the reverse is true when it comes to

<sup>7</sup>Results were qualitatively similar if we used the fiction portions. The age of the Brown corpus makes it less likely to contain modern movie quotes.

<sup>8</sup>We employ Laplace (additive) smoothing with a smoothing parameter of 0.2. The language models’ vocabulary was that of the entire training corpus.

<sup>9</sup>Throughout we obtain part-of-speech tags by using the NLTK maximum entropy tagger with default parameters.

“common language” model		IMDb-only	+Google
lexical	1-gram	61.13%***	59.21%***
	2-gram	59.22%***	57.03%***
	3-gram	59.81%***	58.32%***
syntactic	1-gram	43.60%***	44.77%***
	2-gram	48.31%	47.84%
	3-gram	50.91%	50.92%

Table 3: Distinctiveness: percentage of quote pairs in which the the memorable quote is more distinctive than the non-memorable one according to the respective “common language” model. Significance according to a two-tailed sign test is indicated using \*-notation (\*\*\*)=“ $p < .001$ ”.

syntax: memorable quotes appear to follow the syntactic patterns of “common language” as closely as or more closely than non-memorable quotes. Together, these results suggest that memorable quotes consist of unusual word sequences built on common syntactic scaffolding.

### 3.2 Generality

Another of our hypotheses is that memorable quotes are easier to use outside the specific context in which they were uttered — that is, more “portable” — and therefore exhibit fewer terms that refer to those settings. We use the following syntactic properties as proxies for the generality of a quote:

- **Fewer 3<sup>rd</sup>-person pronouns**, since these commonly refer to a person or object that was introduced earlier in the discourse. Utterances that employ fewer such pronouns are easier to adapt to new contexts, and so will be considered more general.
- **More indefinite articles** like *a* and *an*, since they are more likely to refer to general concepts than definite articles. Quotes with more indefinite articles will be considered more general.
- **Fewer past tense verbs and more present tense verbs**, since the former are more likely to refer to specific previous events. Therefore utterances that employ fewer past tense verbs (and more present tense verbs) will be considered more general.

Table 4 gives the results for each of these four metrics — in each case, we show the percentage of

Generality metric	IMDb-only	+Google
fewer 3 <sup>rd</sup> pers. pronouns	64.37%***	62.93%***
more indef. article	57.21%***	58.23%***
less past tense	57.91%***	59.74%***
more present tense	54.60%***	55.86%***

Table 4: Generality: percentage of quote pairs in which the memorable quote is more general than the non-memorable ones according to the respective metric. Pairs where the metric does not distinguish between the quotes are not considered.

quote pairs for which the memorable quote scores better on the generality metric.

Note that because the issue of generality is a complex one for which there is no straightforward single metric, our approach here is based on several proxies for generality, considered independently; yet, as the results show, all of these point in a consistent direction. It is an interesting open question to develop richer ways of assessing whether a quote has greater generality, in the sense that people intuitively attribute to memorable quotes.

### 3.3 “Memorable” language beyond movies

One of the motivating questions in our analysis is whether there are general principles underlying “memorable language.” The results thus far suggest potential families of such principles. A further question in this direction is whether the notion of memorability can be extended across different domains, and for this we collected (and distribute on our website) 431 phrases that were explicitly designed to be memorable: advertising slogans (e.g., “Quality never goes out of style.”). The focus on slogans is also in keeping with one of the initial motivations in studying memorability, namely, marketing applications — in other words, assessing whether a proposed slogan has features that are consistent with memorable text.

The fact that it’s not clear how to construct a collection of “non-memorable” counterparts to slogans appears to pose a technical challenge. However, we can still use a language-modeling approach to assess whether the textual properties of the slogans are closer to the memorable movie quotes (as one would conjecture) or to the non-memorable movie quotes. Specifically, we train one language model on memorable quotes and another on non-memorable quotes

(Non)memorable language models		Slogans	Newswire
lexical	1-gram	56.15%**	33.77%***
	2-gram	51.51%	25.15%***
	3-gram	52.44%	28.89%***
syntactic	1-gram	73.09%***	68.27%***
	2-gram	64.04%***	50.21%
	3-gram	62.88%***	55.09%***

Table 5: Cross-domain concept of “memorable” language: percentage of slogans that have higher likelihood under the memorable language model than under the non-memorable one (for each of the six language models considered). Rightmost column: for reference, the percentage of newswire sentences that have higher likelihood under the memorable language model than under the non-memorable one.

Generality metric	slogans	mem.	n-mem.
% 3 <sup>rd</sup> pers. pronouns	2.14%	2.16%	3.41%
% indefinite articles	2.68%	2.63%	2.06%
% past tense	14.60%	21.13%	26.69%

Table 6: Slogans are most general when compared to memorable and non-memorable quotes. (%s of 3<sup>rd</sup> pers. pronouns and indefinite articles are relative to all tokens, %s of past tense are relative to all past and present verbs.)

and compare how likely each slogan is to be produced according to these two models. As shown in the middle column of Table 5, we find that slogans are better predicted both lexically and syntactically by the former model. This result thus offers evidence for a concept of “memorable language” that can be applied beyond a single domain.

We also note that the higher likelihood of slogans under a “memorable language” model is not simply occurring for the trivial reason that this model predicts all other large bodies of text better. In particular, the newswire section of the Brown corpus is predicted better at the lexical level by the language model trained on non-memorable quotes.

Finally, Table 6 shows that slogans employ general language, in the sense that for each of our generality metrics, we see a slogans/memorable-quotes/non-memorable quotes spectrum.

### 3.4 Prediction task

We now show how the principles discussed above can provide features for a basic prediction task, corresponding to the task in our human pilot study:

given a pair of quotes, identify the memorable one.

Our first formulation of the prediction task uses a standard bag-of-words model<sup>10</sup>. If there were no information in the textual content of a quote to determine whether it were memorable, then an SVM employing bag-of-words features should perform no better than chance. Instead, though, it obtains 59.67% (10-fold cross-validation) accuracy, as shown in Table 7. We then develop models using features based on the measures formulated earlier in this section: generality measures (the four listed in Table 4); distinctiveness measures (likelihood according to 1, 2, and 3-gram “common language” models at the lexical and part-of-speech level for each quote in the pair, their differences, and pairwise comparisons between them); and similarity-to-slogans measures (likelihood according to 1, 2, and 3-gram slogan-language models at the lexical and part-of-speech level for each quote in the pair, their differences, and pairwise comparisons between them).

Even a relatively small number of distinctiveness features, on their own, improve significantly over the much larger bag-of-words model. When we include additional features based on generality and language-model features measuring similarity to slogans, the performance improves further (last line of Table 7).

Thus, the main conclusion from these prediction tasks is that abstracting notions such as distinctiveness and generality can produce relatively streamlined models that outperform much heavier-weight bag-of-words models, and can suggest steps toward approaching the performance of human judges who — very much unlike our system — have the full cultural context in which movies occur at their disposal.

### 3.5 Other characteristics

We also made some auxiliary observations that may be of interest. Specifically, we find differences in letter and sound distribution (e.g., memorable quotes — after curse-word removal — use significantly more “front sounds” (labials or front vowels such as represented by the letter *i*) and significantly fewer “back sounds” such as the one represented by *u*),<sup>11</sup>

<sup>10</sup>We discarded terms appearing fewer than 10 times.

<sup>11</sup>These findings may relate to marketing research on *sound symbolism* [7, 19, 40].

Feature set	# feats	Accuracy
bag of words	962	59.67%
distinctiveness	24	62.05%*
generality	4	56.70%
slogan sim.	24	58.30%
<i>all three types together</i>	52	64.27%**

Table 7: Prediction: SVM 10-fold cross validation results using the respective feature sets. Random baseline accuracy is 50%. Accuracies statistically significantly greater than bag-of-words according to a two-tailed t-test are indicated with \*( $p < .05$ ) and \*\*( $p < .01$ ).

word complexity (e.g., memorable quotes use words with significantly more syllables) and phrase complexity (e.g., memorable quotes use fewer coordinating conjunctions). The latter two are in line with our distinctiveness hypothesis.

#### 4 A long time ago, in a galaxy far, far away

How an item’s linguistic form affects the reaction it generates has been studied in several contexts, including evaluations of product reviews [9], political speeches [12], on-line posts [13], scientific papers [14], and retweeting of Twitter posts [36]. We use a different set of features, abstracting the notions of distinctiveness and generality, in order to focus on these higher-level aspects of phrasing rather than on particular lower-level features.

Related to our interest in distinctiveness, work in advertising research has studied the effect of syntactic complexity on recognition and recall of slogans [5, 6, 24]. There may also be connections to Von Restorff’s *isolation effect* Hunt [17], which asserts that when all but one item in a list are similar in some way, memory for the different item is enhanced.

Related to our interest in generality, Knapp et al. [20] surveyed subjects regarding memorable messages or pieces of advice they had received, finding that the ability to be applied to multiple concrete situations was an important factor.

Memorability, although distinct from “memorizability”, relates to short- and long-term recall. Thorn and Page [34] survey sub-lexical, lexical, and semantic attributes affecting short-term memorability of lexical items. Studies of verbatim recall have also considered the task of distinguishing an exact quote from close paraphrases [3]. Investigations of long-term recall have included studies of culturally signif-

icant passages of text [29] and findings regarding the effect of rhetorical devices of alliterative [4], “rhythmic, poetic, and thematic constraints” [18, 26].

Finally, there are complex connections between humor and memory [32], which may lead to interactions with computational humor recognition [25].

#### 5 I think this is the beginning of a beautiful friendship.

Motivated by the broad question of what kinds of information achieve widespread public awareness, we studied the the effect of phrasing on a quote’s memorability. A challenge is that quotes differ not only in how they are worded, but also in who said them and under what circumstances; to deal with this difficulty, we constructed a controlled corpus of movie quotes in which lines deemed memorable are paired with non-memorable lines spoken by the same character at approximately the same point in the same movie. After controlling for context and situation, memorable quotes were still found to exhibit, *on average* (there will always be individual exceptions), significant differences from non-memorable quotes in several important respects, including measures capturing distinctiveness and generality. Our experiments with slogans show how the principles we identify can extend to a different domain.

Future work may lead to applications in marketing, advertising and education [4]. Moreover, the subtle nature of memorability, and its connection to research in psychology, suggests a range of further research directions. We believe that the framework developed here can serve as the basis for further computational studies of the process by which information takes hold in the public consciousness, and the role that language effects play in this process.

**My mother thanks you. My father thanks you. My sister thanks you. And I thank you:** Rebecca Hwa, Evie Kleinberg, Diana Minculescu, Alex Niculescu-Mizil, Jennifer Smith, Benjamin Zimmer, and the anonymous reviewers for helpful discussions and comments; our annotators Steven An, Lars Backstrom, Eric Baumer, Jeff Chadwick, Evie Kleinberg, and Myle Ott; and the makers of Cepacol, Robitussin, and Sudafed, whose products got us through the submission deadline. This paper is based upon work supported in part by NSF grants IIS-0910664, IIS-1016099, Google, and Yahoo!

## References

- [1] Eytan Adar, Li Zhang, Lada A. Adamic, and Rajan M. Lukose. Implicit structure and the dynamics of blogspace. In *Workshop on the Weblogging Ecosystem*, 2004.
- [2] Lars Backstrom, Dan Huttenlocher, Jon Kleinberg, and Xiangyang Lan. Group formation in large social networks: Membership, growth, and evolution. In *Proceedings of KDD*, 2006.
- [3] Elizabeth Bates, Walter Kintsch, Charles R. Fletcher, and Vittoria Giuliani. The role of pronominalization and ellipsis in texts: Some memory experiments. *Journal of Experimental Psychology: Human Learning and Memory*, 6(6):676–691, 1980.
- [4] Frank Boers and Seth Lindstromberg. Finding ways to make phrase-learning feasible: The mnemonic effect of alliteration. *System*, 33(2): 225–238, 2005.
- [5] Samuel D. Bradley and Robert Meeds. Surface-structure transformations and advertising slogans: The case for moderate syntactic complexity. *Psychology and Marketing*, 19: 595–619, 2002.
- [6] Robert Chamberlee, Robert Gilmore, Gloria Thomas, and Gary Soldow. When copy complexity can help ad readership. *Journal of Advertising Research*, 33(3):23–23, 1993.
- [7] John Colapinto. Famous names. *The New Yorker*, pages 38–43, 2011.
- [8] Cristian Danescu-Niculescu-Mizil and Lillian Lee. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics*, 2011.
- [9] Cristian Danescu-Niculescu-Mizil, Gueorgi Kossinets, Jon Kleinberg, and Lillian Lee. How opinions are received by online communities: A case study on Amazon.com helpfulness votes. In *Proceedings of WWW*, pages 141–150, 2009.
- [10] Stuart Fischhoff, Esmeralda Cardenas, Angela Hernandez, Corey Wyatt, Jared Young, and Rachel Gordon. Popular movie quotes: Reflections of a people and a culture. In *Annual Convention of the American Psychological Association*, 2000.
- [11] Daniel Gruhl, R. Guha, David Liben-Nowell, and Andrew Tomkins. Information diffusion through blogspace. *Proceedings of WWW*, pages 491–501, 2004.
- [12] Marco Guerini, Carlo Strapparava, and Oliviero Stock. Trusting politicians’ words (for persuasive NLP). In *Proceedings of CICLing*, pages 263–274, 2008.
- [13] Marco Guerini, Carlo Strapparava, and Gözde Özbal. Exploring text virality in social networks. In *Proceedings of ICWSM (poster)*, 2011.
- [14] Marco Guerini, Alberto Pepe, and Bruno Lepri. Do linguistic style and readability of scientific abstracts affect their virality? In *Proceedings of ICWSM*, 2012.
- [15] Richard Jackson Harris, Abigail J. Werth, Kyle E. Bures, and Chelsea M. Bartel. Social movie quoting: What, why, and how? *Ciencias Psicológicas*, 2(1):35–45, 2008.
- [16] Chip Heath, Chris Bell, and Emily Steinberg. Emotional selection in memes: The case of urban legends. *Journal of Personality*, 81(6): 1028–1041, 2001.
- [17] R. Reed Hunt. The subtlety of distinctiveness: What von Restorff really did. *Psychonomic Bulletin & Review*, 2(1):105–112, 1995.
- [18] Ira E. Hyman Jr. and David C. Rubin. Memorabilia: A naturalistic study of long-term memory. *Memory & Cognition*, 18(2):205–214, 1990.
- [19] Richard R. Klink. Creating brand names with meaning: The use of sound symbolism. *Marketing Letters*, 11(1):5–20, 2000.
- [20] Mark L. Knapp, Cynthia Stohl, and Kathleen K. Reardon. “Memorable” messages. *Journal of Communication*, 31(4):27–41, 1981.
- [21] Henry Kučera and W. Nelson Francis. *Computational analysis of present-day American English*. Dartmouth Publishing Group, 1967.



- [22] Jure Leskovec, Lada Adamic, and Bernardo Huberman. The dynamics of viral marketing. *ACM Transactions on the Web*, 1(1), May 2007.
- [23] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *Proceedings of KDD*, pages 497–506, 2009.
- [24] Tina M. Lowrey. The relation between script complexity and commercial memorability. *Journal of Advertising*, 35(3):7–15, 2006.
- [25] Rada Mihalcea and Carlo Strapparava. Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence*, 22(2):126–142, 2006.
- [26] Milman Parry and Adam Parry. *The making of Homeric verse: The collected papers of Milman Parry*. Clarendon Press, Oxford, 1971.
- [27] Everett Rogers. *Diffusion of Innovations*. Free Press, fourth edition, 1995.
- [28] Daniel M. Romero, Brendan Meeder, and Jon Kleinberg. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on Twitter. *Proceedings of WWW*, pages 695–704, 2011.
- [29] David C. Rubin. Very long-term memory for prose and verse. *Journal of Verbal Learning and Verbal Behavior*, 16(5):611–621, 1977.
- [30] Nathan Schneider, Rebecca Hwa, Philip Gianfortoni, Dipanjan Das, Michael Heilman, Alan W. Black, Frederick L. Crabbe, and Noah A. Smith. Visualizing topical quotations over time to understand news discourse. Technical Report CMU-LTI-01-103, CMU, 2010.
- [31] David Strang and Sarah Soule. Diffusion in organizations and social movements: From hybrid corn to poison pills. *Annual Review of Sociology*, 24:265–290, 1998.
- [32] Hannah Summerfelt, Louis Lippman, and Ira E. Hyman Jr. The effect of humor on memory: Constrained by the pun. *The Journal of General Psychology*, 137(4), 2010.
- [33] Eric Sun, Itamar Rosenn, Cameron Marlow, and Thomas M. Lento. Gesundheit! Modeling contagion through Facebook News Feed. In *Proceedings of ICWSM*, 2009.
- [34] Annabel Thorn and Mike Page. *Interactions Between Short-Term and Long-Term Memory in the Verbal Domain*. Psychology Press, 2009.
- [35] Louis L. Thurstone. A law of comparative judgment. *Psychological Review*, 34(4):273–286, 1927.
- [36] Oren Tsur and Ari Rappoport. What’s in a Hashtag? Content based prediction of the spread of ideas in microblogging communities. In *Proceedings of WSDM*, 2012.
- [37] Fang Wu, Bernardo A. Huberman, Lada A. Adamic, and Joshua R. Tyler. Information flow in social groups. *Physica A: Statistical and Theoretical Physics*, 337(1-2):327–335, 2004.
- [38] Shaomei Wu, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Who says what to whom on Twitter. In *Proceedings of WWW*, 2011.
- [39] Jaewon Yang and Jure Leskovec. Patterns of temporal variation in online media. In *Proceedings of WSDM*, 2011.
- [40] Eric Yorkston and Geeta Menon. A sound idea: Phonetic effects of brand names on consumer judgments. *Journal of Consumer Research*, 31(1):43–51, 2004.

# Modeling the Translation of Predicate-Argument Structure for SMT

Deyi Xiong, Min Zhang,\* Haizhou Li

Human Language Technology

Institute for Infocomm Research

1 Fusionopolis Way, #21-01 Connexis, Singapore 138632

{dyxiong, mzhang, hli}@i2r.a-star.edu.sg

## Abstract

Predicate-argument structure contains rich semantic information of which statistical machine translation hasn't taken full advantage. In this paper, we propose two discriminative, feature-based models to exploit predicate-argument structures for statistical machine translation: 1) a predicate translation model and 2) an argument reordering model. The predicate translation model explores lexical and semantic contexts surrounding a verbal predicate to select desirable translations for the predicate. The argument reordering model automatically predicts the moving direction of an argument relative to its predicate after translation using semantic features. The two models are integrated into a state-of-the-art phrase-based machine translation system and evaluated on Chinese-to-English translation tasks with large-scale training data. Experimental results demonstrate that the two models significantly improve translation accuracy.

## 1 Introduction

Recent years have witnessed increasing efforts towards integrating predicate-argument structures into statistical machine translation (SMT) (Wu and Fung, 2009b; Liu and Gildea, 2010). In this paper, we take a step forward by introducing a novel approach to incorporate such semantic structures into SMT. Given a source side predicate-argument structure, we attempt to translate each semantic frame (predicate and its associated arguments) into an appropriate target string. We believe that the translation of predicates and reordering of arguments are the two central

issues concerning the transfer of predicate-argument structure across languages.

Predicates<sup>1</sup> are essential elements in sentences. Unfortunately they are usually neither correctly translated nor translated at all in many SMT systems according to the error study by Wu and Fung (2009a). This suggests that conventional lexical and phrasal translation models adopted in those SMT systems are not sufficient to correctly translate predicates in source sentences. Thus we propose a discriminative, feature-based **predicate translation model** that captures not only lexical information (i.e., surrounding words) but also high-level semantic contexts to correctly translate predicates.

Arguments contain information for questions of *who*, *what*, *when*, *where*, *why*, and *how* in sentences (Xue, 2008). One common error in translating arguments is about their reorderings: arguments are placed at incorrect positions after translation. In order to reduce such errors, we introduce a discriminative **argument reordering model** that uses the position of a predicate as the reference axis to estimate positions of its associated arguments on the target side. In this way, the model predicts moving directions of arguments relative to their predicates with semantic features.

We integrate these two discriminative models into a state-of-the-art phrase-based system. Experimental results on large-scale Chinese-to-English translation show that both models are able to obtain significant improvements over the baseline. Our analysis on system outputs further reveals that they can indeed help reduce errors in predicate translations and argument reorderings.

\*Corresponding author

<sup>1</sup>We only consider verbal predicates in this paper.

The paper is organized as follows. In Section 2, we will introduce related work and show the significant differences between our models and previous work. In Section 3 and 4, we will elaborate the proposed predicate translation model and argument reordering model respectively, including details about modeling, features and training procedure. Section 5 will introduce how to integrate these two models into SMT. Section 6 will describe our experiments and results. Section 7 will empirically discuss how the proposed models improve translation accuracy. Finally we will conclude with future research directions in Section 8.

## 2 Related Work

Predicate-argument structures (PAS) are explored for SMT on both the source and target side in some previous work. As PAS analysis widely employs global and sentence-wide features, it is computationally expensive to integrate target side predicate-argument structures into the dynamic programming style of SMT decoding (Wu and Fung, 2009b). Therefore they either postpone the integration of target side PASs until the whole decoding procedure is completed (Wu and Fung, 2009b), or directly project semantic roles from the source side to the target side through word alignments during decoding (Liu and Gildea, 2010).

There are other previous studies that explore only source side predicate-argument structures. Komachi and Matsumoto (2006) reorder arguments in source language (Japanese) sentences using heuristic rules defined on source side predicate-argument structures in a pre-processing step. Wu et al. (2011) automate this procedure by automatically extracting reordering rules from predicate-argument structures and applying these rules to reorder source language sentences. Aziz et al. (2011) incorporate source language semantic role labels into a tree-to-string SMT system.

Although we also focus on source side predicate-argument structures, our models differ from the previous work in two main aspects: 1) we propose two separate discriminative models to exploit predicate-argument structures for predicate translation and argument reordering respectively; 2) we consider argument reordering as an argument movement (rel-

ative to its predicate) prediction problem and use a discriminatively trained classifier for such predictions.

Our predicate translation model is also related to previous discriminative lexicon translation models (Berger et al., 1996; Venkatapathy and Bangalore, 2007; Mauser et al., 2009). While previous models predict translations for all words in vocabulary, we only focus on verbal predicates. This will tremendously reduce the amount of training data required, which usually is a problem in discriminative lexicon translation models (Mauser et al., 2009). Furthermore, the proposed translation model also differs from previous lexicon translation models in that we use both lexical and semantic features. Our experimental results show that semantic features are able to further improve translation accuracy.

## 3 Predicate Translation Model

In this section, we present the features and the training process of the predicate translation model.

### 3.1 Model

Following the context-dependent word models in (Berger et al., 1996), we propose a discriminative predicate translation model. The essential component of our model is a maximum entropy classifier  $p_t(e|\mathcal{C}(v))$  that predicts the target translation  $e$  for a verbal predicate  $v$  given its surrounding context  $\mathcal{C}(v)$ . The classifier can be formulated as follows.

$$p_t(e|\mathcal{C}(v)) = \frac{\exp(\sum_i \theta_i f_i(e, \mathcal{C}(v)))}{\sum_{e'} \exp(\sum_i \theta_i f_i(e', \mathcal{C}(v)))} \quad (1)$$

where  $f_i$  are binary features,  $\theta_i$  are weights of these features. Given a source sentence which contains  $N$  verbal predicates  $\{v_i\}_1^N$ , our predicate translation model  $M_t$  can be denoted as

$$M_t = \prod_{i=1}^N p_t(e_{v_i}|\mathcal{C}(v_i)) \quad (2)$$

Note that we do not restrict the target translation  $e$  to be a single word. We allow  $e$  to be a phrase of length up to 4 words so as to capture multi-word translations for a verbal predicate. For example, a Chinese verb “发行(issue)” can be translated as “to be issued” or “have issued” with modality words.

This will increase the number of classes to be predicted by the maximum entropy classifier. But according to our observation, it is still computationally tractable (see Section 3.3). If a verbal predicate is not translated, we set  $e = \text{NULL}$  so that we can also capture null translations for verbal predicates.

### 3.2 Features

The apparent advantage of discriminative lexicon translation models over generative translation models (e.g., conventional lexical translation model as described in (Koehn et al., 2003)) is that discriminative models allow us to integrate richer contexts (lexical, syntactic or semantic) into target translation prediction. We use two kinds of features to predict translations for verbal predicates: 1) lexical features and 2) semantic features. All features are in the following binary form.

$$f(e, \mathcal{C}(v)) = \begin{cases} 1, & \text{if } e = \clubsuit \text{ and } \mathcal{C}(v). \heartsuit = \spadesuit \\ 0, & \text{else} \end{cases} \quad (3)$$

where the symbol  $\clubsuit$  is a placeholder for a possible target translation (up to 4 words), the symbol  $\heartsuit$  indicates a contextual (lexical or semantic) element for the verbal predicate  $v$ , and the symbol  $\spadesuit$  represents the value of  $\heartsuit$ .

**Lexical Features:** The lexical element  $\heartsuit$  is extracted from the surrounding words of verbal predicate  $v$ . We use the preceding 3 words and the succeeding 3 words to define the lexical context for the verbal predicate  $v$ . Therefore  $\heartsuit \in \{w_{-3}, w_{-2}, w_{-1}, v, w_1, w_2, w_3\}$ .

**Semantic Features:** The semantic element  $\heartsuit$  is extracted from the surrounding arguments of verbal predicate  $v$ . In particular, we define a semantic window centered at the verbal predicate with 6 arguments  $\{A_{-3}, A_{-2}, A_{-1}, A_1, A_2, A_3\}$  where  $A_{-3} - A_{-1}$  are arguments on the left side of  $v$  while  $A_1 - A_3$  are those on the right side. Different verbal predicates have different number of arguments in different linguistic scenarios. We observe on our training data that the number of arguments for 96.5% verbal predicates on each side (left/right) is not larger than 3. Therefore the defined 6-argument semantic window is sufficient to describe argument contexts for predicates.

For each argument  $A_i$  in the defined seman-

$f(e, \mathcal{C}(v)) = 1$  **if and only if**

---

$e = \text{adjourn}$  and  $\mathcal{C}(v).A_{-3}^h = \text{安理会}$   
 $e = \text{adjourn}$  and  $\mathcal{C}(v).A_{-1}^r = \text{ARGM-TMP}$   
 $e = \text{adjourn}$  and  $\mathcal{C}(v).A_1^h = \text{天}$   
 $e = \text{adjourn}$  and  $\mathcal{C}(v).A_2^r = \text{null}$   
 $e = \text{adjourn}$  and  $\mathcal{C}(v).A_3^h = \text{null}$

Table 1: Semantic feature examples.

tic window, we use its semantic role (i.e., ARG0, ARGM-TMP and so on)  $A_i^r$  and head word  $A_i^h$  to define semantic context elements  $\heartsuit$ . If an argument  $A_i$  does not exist for the verbal predicate  $v$ <sup>2</sup>, we set the value of both  $A_i^r$  and  $A_i^h$  to null.

Figure 1 shows a Chinese sentence with its predicate-argument structure and English translation. The verbal predicate “**休会/adjourn**” (in bold) has 4 arguments: one in an ARG0 agent role, one in an ARGM-ADV adverbial modifier role, one in an ARGM-TMP temporal modifier role and the last one in an ARG1 patient role. Table 1 shows several semantic feature examples of this verbal predicate.

### 3.3 Training

In order to train the discriminative predicate translation model, we first parse source sentences and labeled semantic roles for all verbal predicates (see details in Section 6.1) in our word-aligned bilingual training data. Then we extract all training events for verbal predicates which occur at least 10 times in the training data. A training event for a verbal predicate  $v$  consists of all contextual elements  $\mathcal{C}(v)$  (e.g.,  $w_1, A_1^h$ ) defined in the last section and the target translation  $e$ . Using these events, we train one maximum entropy classifier per verbal predicate (16,121 verbs in total) via the off-the-shelf MaxEnt toolkit<sup>3</sup>. We perform 100 iterations of the L-BFGS algorithm implemented in the training toolkit for each verbal predicate with both Gaussian prior and event cutoff set to 1 to avoid overfitting. After event cutoff, we have an average of 140 classes (target translations) per verbal predicate with the maximum number of classes being 9,226. The training takes an average of 52.6 seconds per verb. In order to expedite the train-

<sup>2</sup>For example, the verb  $v$  has only two arguments on its left side. Thus argument  $A_{-3}$  does not exist.

<sup>3</sup>Available at: [http://homepages.inf.ed.ac.uk/lzhang10/maxent\\_toolkit.html](http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html)

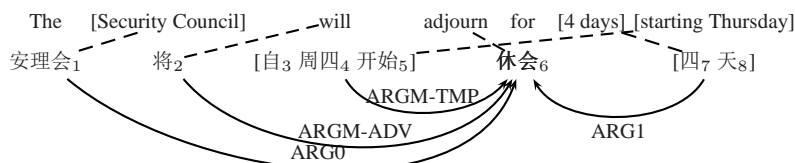


Figure 1: An example of predicate-argument structure in Chinese and its aligned English translation. The bold word in Chinese is the verbal predicate. The subscripts on the Chinese sentence show the indexes of words from left to right.

ing, we run the training toolkit in a parallel manner.

## 4 Argument Reordering Model

In this section we introduce the discriminative argument reordering model, features and the training procedure.

### 4.1 Model

Since the predicate determines what arguments are involved in its semantic frame and semantic frames tend to be cohesive across languages (Fung et al., 2006), the movements of predicate and its arguments across translations are like the motions of a planet and its satellites. Therefore we consider the reordering of an argument as the motion of the argument relative to its predicate. In particular, we use the position of the predicate as the reference axis. The motion of associated arguments relative to the reference axis can be roughly divided into 3 categories<sup>4</sup>: 1) no change across languages (NC); 2) moving from the left side of its predicate to the right side of the predicate after translation (L2R); and 3) moving from the right side of its predicate to the left side of the predicate after translation (R2L).

Let’s revisit Figure 1. The ARG0, ARGM-ADV and ARG1 are located at the same side of their predicate after being translated into English, therefore the reordering category of these three arguments is assigned as “NC”. The ARGM-TMP is moved from the left side of “休会/adjourn” to the right side of “adjourn” after translation, thus its reordering category is L2R.

In order to predict the reordering category for an argument, we propose a discriminative argument reordering model that uses a maximum en-

<sup>4</sup>Here we assume that the translations of arguments are not interrupted by their predicates, other arguments or any words outside the arguments in question. We leave for future research the task of determining whether arguments should be translated as a unit or not.

ropy classifier to calculate the reordering category  $m \in \{\text{NC}, \text{L2R}, \text{R2L}\}$  for an argument  $A$  as follows.

$$p_r(m|\mathcal{C}(A)) = \frac{\exp(\sum_i \theta_i f_i(m, \mathcal{C}(A)))}{\sum_{m'} \exp(\sum_i \theta_i f_i(m', \mathcal{C}(A)))} \quad (4)$$

where  $\mathcal{C}(A)$  indicates the surrounding context of  $A$ . The features  $f_i$  will be introduced in the next section. We assume that motions of arguments are independent on each other. Given a source sentence with labeled arguments  $\{A_i\}_1^N$ , our discriminative argument reordering model  $M_r$  is formulated as

$$M_r = \prod_{i=1}^N p_r(m_{A_i}|\mathcal{C}(A_i)) \quad (5)$$

### 4.2 Features

The features  $f_i$  used in the argument reordering model still takes the binary form as in Eq. (3). Table 2 shows the features that are used in the argument reordering model. We extract features from both the source and target side. On the source side, the features include the verbal predicate, the semantic role of the argument, the head word and the boundary words of the argument. On the target side, the translation of the verbal predicate, the translation of the head word of the argument, as well as the boundary words of the translation of the argument are used as features.

### 4.3 Training

To train the argument reordering model, we first extract features defined in the last section from our bilingual training data where source sentences are annotated with predicate-argument structures. We also study the distribution of argument reordering categories (i.e., NC, L2R and R2L) in the training data, which is shown in Table 3. Most arguments, accounting for 82.43%, are on the same side of their verbal predicates after translation. The remaining



	Features of an argument $A$ for reordering
<b>src</b>	its verbal predicate $A^p$
	its semantic role $A^r$
	its head word $A^h$
	the leftmost word of $A$
	the rightmost word of $A$
<b>tgt</b>	the translation of $A^p$
	the translation of $A^h$
	the leftmost word of the translation of $A$
	the rightmost word of the translation of $A$

Table 2: Features adopted in the argument reordering model.

Reordering Category	Percent
NC	82.43%
L2R	11.19%
R2L	6.38%

Table 3: Distribution of argument reordering categories in the training data.

arguments (17.57%) are moved either from the left side of their predicates to the right side after translation (accounting for 11.19%) or from the right side to the left side of their translated predicates (accounting for 6.38%).

After all features are extracted, we use the maximum entropy toolkit in Section 3.3 to train the maximum entropy classifier as formulated in Eq. (4). We perform 100 iterations of L-BFGS.

## 5 Integrating the Two Models into SMT

In this section, we elaborate how to integrate the two models into phrase-based SMT. In particular, we integrate the models into a phrase-based system which uses bracketing transduction grammars (BTG) (Wu, 1997) for phrasal translation (Xiong et al., 2006). Since the system is based on a CKY-style decoder, the integration algorithms introduced here can be easily adapted to other CKY-based decoding systems such as the hierarchical phrasal system (Chiang, 2007).

### 5.1 Integrating the Predicate Translation Model

It is straightforward to integrate the predicate translation model into phrase-based SMT (Koehn et al.,

2003; Xiong et al., 2006). We maintain word alignments for each phrase pair in the phrase table. Given a source sentence with its predicate-argument structure, we detect all verbal predicates and load trained predicate translation classifiers for these verbs. Whenever a hypothesis covers a new verbal predicate  $v$ , we find the target translation  $e$  for  $v$  through word alignments and then calculate its translation probability  $p_t(e|\mathcal{C}(v))$  according to Eq. (1).

The predicate translation model (as formulated in Eq. (2)) is integrated into the whole log-linear model just like the conventional lexical translation model in phrase-based SMT (Koehn et al., 2003). The two models are independently estimated but complementary to each other. While the lexical translation model calculates the probability of a verbal predicate being translated given its local lexical context, the discriminative predicate translation model is able to employ both lexical and semantic contexts to predict translations for verbs.

### 5.2 Integrating the Argument Reordering Model

Before we introduce the integration algorithm for the argument reordering model, we define two functions  $\mathcal{A}$  and  $\mathcal{N}$  on a source sentence and its predicate-argument structure  $\tau$  as follows.

- $\mathcal{A}(i, j, \tau)$ : from the predicate-argument structure  $\tau$ , the function finds all predicate-argument pairs which are completely located within the span from source word  $i$  to  $j$ . For example, in Figure 1,  $\mathcal{A}(3, 6, \tau) = \{(\text{休会}, \text{ARGM-TMP})\}$  while  $\mathcal{A}(2, 3, \tau) = \{\}$ ,  $\mathcal{A}(1, 5, \tau) = \{\}$  because the verbal predicate “休会” is located outside the span (2,3) and (1,5).
- $\mathcal{N}(i, k, j, \tau)$ : the function finds all predicate-argument pairs that cross the two neighboring spans  $(i, k)$  and  $(k+1, j)$ . It can be formulated as  $\mathcal{A}(i, j, \tau) - (\mathcal{A}(i, k, \tau) \cup \mathcal{A}(k+1, j, \tau))$ .

We then define another function  $\mathcal{P}_r$  to calculate the argument reordering model probability on all arguments which are found by the previous two functions  $\mathcal{A}$  and  $\mathcal{N}$  as follows.

$$\mathcal{P}_r(\mathcal{B}) = \prod_{A \in \mathcal{B}} p_r(m_A | \mathcal{C}(A)) \quad (6)$$

where  $\mathcal{B}$  denotes either  $\mathcal{A}$  or  $\mathcal{N}$ .

Following (Chiang, 2007), we describe the algorithm in a deductive system. It is shown in Figure 2. The algorithm integrates the argument reordering model into a CKY-style decoder (Xiong et al., 2006). The item  $[X, i, j]$  denotes a BTG node  $X$  spanning from  $i$  to  $j$  on the source side. For notational convenience, we only show the argument reordering model probability for each item, ignoring all other sub-model probabilities such as the language model probability. The Eq. (7) shows how we calculate the argument reordering model probability when a lexical rule is applied to translate a source phrase  $c$  to a target phrase  $e$ . The Eq. (8) shows how we compute the argument reordering model probability for a span  $(i, j)$  in a dynamic programming manner when a merging rule is applied to combine its two sub-spans in a straight ( $X \rightarrow [X_1, X_2]$ ) or inverted order ( $X \rightarrow \langle X_1, X_2 \rangle$ ). We directly use the probabilities  $\mathcal{P}_r(\mathcal{A}(i, k, \tau))$  and  $\mathcal{P}_r(\mathcal{A}(k + 1, j, \tau))$  that have been already obtained for the two sub-spans  $(i, k)$  and  $(k + 1, j)$ . In this way, we only need to calculate the probability  $\mathcal{P}_r(\mathcal{N}(i, k, j, \tau))$  for predicate-argument pairs that cross the two sub-spans.

## 6 Experiments

In this section, we present our experiments on Chinese-to-English translation tasks, which are trained with large-scale data. The experiments are aimed at measuring the effectiveness of the proposed discriminative predicate translation model and argument reordering model.

### 6.1 Setup

The baseline system is the BTG-based phrasal system (Xiong et al., 2006). Our training corpora<sup>5</sup> consist of 3.8M sentence pairs with 96.9M Chinese words and 109.5M English words. We ran GIZA++ on these corpora in both directions and then applied the “grow-diag-final” refinement rule to obtain word alignments. We then used all these word-aligned corpora to generate our phrase table. Our 5-gram language model was trained on the Xinhua section of the English Gigaword corpus (306 million words)

<sup>5</sup>The corpora include LDC2004E12, LDC2004T08, LDC2005T10, LDC2003E14, LDC2002E18, LDC2005T06, LDC2003E07 and LDC2004T07.

using the SRILM toolkit (Stolcke, 2002) with modified Kneser-Ney smoothing.

To train the proposed predicate translation model and argument reordering model, we first parsed all source sentences using the Berkeley Chinese parser (Petrov et al., 2006) and then ran the Chinese semantic role labeler<sup>6</sup> (Li et al., 2010) on all source parse trees to annotate semantic roles for all verbal predicates. After we obtained semantic roles on the source side, we extracted features as described in Section 3.2 and 4.2 and used these features to train our two models as described in Section 3.3 and 4.3.

We used the NIST MT03 evaluation test data as our development set, and the NIST MT04, MT05 as the test sets. We adopted the case-insensitive BLEU-4 (Papineni et al., 2002) as the evaluation metric. Statistical significance in BLEU differences was tested by paired bootstrap re-sampling (Koehn, 2004).

### 6.2 Results

Our first group of experiments is to investigate whether the predicate translation model is able to improve translation accuracy in terms of BLEU and whether semantic features are useful. The experimental results are shown in Table 4. From the table, we have the following two observations.

- The proposed predicate translation models achieve an average improvement of 0.57 BLEU points across the two NIST test sets when all features (lex+sem) are used. Such an improvement is statistically significant ( $p < 0.01$ ). According to our statistics, there are 5.07 verbal predicates per sentence in NIST04 and 4.76 verbs per sentence in NIST05, which account for 18.02% and 16.88% of all words in NIST04 and 05 respectively. This shows that not only verbal predicates are semantically important, they also form a major part of the sentences. Therefore, whether verbal predicates are translated correctly or not has a great impact on the translation accuracy of the whole sentence<sup>7</sup>.

<sup>6</sup>Available at: <http://nlp.suda.edu.cn/~jhli/>.

<sup>7</sup>The example in Table 6 shows that the translations of verbs even influences reorderings and translations of neighboring words.

$$\frac{X \rightarrow c/e}{[X, i, j] : \mathcal{P}_r(\mathcal{A}(i, j, \tau))} \quad (7)$$

$$\frac{X \rightarrow [X_1, X_2] \text{ or } \langle X_1, X_2 \rangle \quad [X_1, i, k] : \mathcal{P}_r(\mathcal{A}(i, k, \tau)) \quad [X_2, k+1, j] : \mathcal{P}_r(\mathcal{A}(k+1, j, \tau))}{[X, i, j] : \mathcal{P}_r(\mathcal{A}(i, k, \tau)) \cdot \mathcal{P}_r(\mathcal{A}(k+1, j, \tau)) \cdot \mathcal{P}_r(\mathcal{N}(i, k, j, \tau))} \quad (8)$$

Figure 2: Integrating the argument reordering model into a BTG-style decoder.

Model	NIST04	NIST05
Base	35.52	33.80
Base+PTM (lex)	35.71+	34.09+
Base+PTM (lex+sem)	36.10++**	34.35++*

Table 4: Effects of the proposed predicate translation model (PTM). PTM (lex): predicate translation model with lexical features; PTM (lex+sem): predicate translation model with both lexical and semantic features; +/++: better than the baseline ( $p < 0.05/0.01$ ). \*\*/\*: better than Base+PTM (lex) ( $p < 0.05/0.01$ ).

Model	NIST04	NIST05
Base	35.52	33.80
Base+ARM	35.82++	34.29++
Base+ARM+PTM	36.19++	34.72++

Table 5: Effects of the proposed argument reordering model (ARM) and the combination of ARM and PTM. ++: better than the baseline ( $p < 0.01$ ).

- When we integrate both lexical and semantic features (lex+sem) described in Section 3.2, we obtain an improvement of about 0.33 BLEU points over the system where only lexical features (lex) are used. Such a gain, which is statistically significant, confirms the effectiveness of semantic features.

Our second group of experiments is to validate whether the argument reordering model is capable of improving translation quality. Table 5 shows the results. We obtain an average improvement of 0.4 BLEU points on the two test sets over the baseline when we incorporate the proposed argument reordering model into our system. The improvements on the two test sets are both statistically significant ( $p < 0.01$ ).

Finally, we integrate both the predicate translation model and argument reordering model into the final system. The two models collectively achieve an im-

provement of up to 0.92 BLEU points over the baseline, which is shown in Table 5.

## 7 Analysis

In this section, we conduct some case studies to show how the proposed models improve translation accuracy by looking into the differences that they make on translation hypotheses.

Table 6 displays a translation example which shows the difference between the baseline and the system enhanced with the predicate translation model. There are two verbal predicates “赶往/head to” and “参加/attend” in the source sentence. In order to get the most appropriate translations for these two verbal predicates, we should adopt different ways to translate them. The former should be translated as a corresponding verb word or phrase while the latter into a preposition word “for”. Unfortunately, the baseline incorrectly translates the two verbs. Furthermore, such translation errors even result in undesirable reorderings of neighboring words “伯利恒/Bethlehem and “弥撒/mass”. This indicates that verbal predicate translation errors may lead to more errors, such as inappropriate reorderings or lexical choices for neighboring words. On the contrary, we can see that our predicate translation model is able to help select appropriate words for both verbs. The correct translations of these two verbs also avoid incorrect reorderings of neighboring words.

Table 7 shows another example to demonstrate how the argument reordering model improve reorderings. The verbal predicate “进行/carry out” has three arguments, ARG0, ARG-ADV and ARG1. The ARG1 argument should be moved from the right side of the predicate to its left side after translation. The ARG0 argument can either stay on the left side or move to right side of the predicate. Ac-



Base	[数千] 信徒 赶往 伯利恒 参加 [平安夜] 弥撒 [thousands of] followers to Mass in Bethlehem [Christmas Eve]
Base+PTM	[数千] 信徒 赶往 伯利恒 参加 [平安夜] 弥撒 [thousands of] devotees [rushed to] Bethlehem for [Christmas Eve] mass
Ref	thousands of worshippers head to Bethlehem for Christmas Midnight mass

Table 6: A translation example showing the difference between the baseline and the system with the predicate translation model (PTM). Phrase alignments in the two system outputs are shown with dashed lines. Chinese words in bold are verbal predicates.

PAS	
Base	[有关这] 套 灾难 [警告系统] 还要 [进行更多] [重要的磋商] the more [important consultations] also set disaster [warning system]
Base+ARM	有关 [这套] 灾难 [警告系统] [还要进行] [更多] [重要的磋商] more [important consultations] on [such a] disaster [warning system] [should be carried out]
Ref	more important discussions will be held on the disaster warning system

Table 7: A translation example showing the difference between the baseline and the system with the argument reordering model (ARM). The predicate-argument structure (PAS) of the source sentence is also displayed in the first row.

According to the phrase alignments of the baseline, we clearly observe three serious translation errors: 1) the ARG0 argument is translated into separate groups which are not adjacent on the target side; 2) the predicate is not translated at all; and 3) the ARG1 argument is not moved to the left side of the predicate after translation. All of these 3 errors are avoided in the Base+ARM system output as a result of the argument reordering model that correctly identifies arguments and moves them in the right directions.

## 8 Conclusions and Future Work

We have presented two discriminative models to incorporate source side predicate-argument structures into SMT. The two models have been integrated into a phrase-based SMT system and evaluated on Chinese-to-English translation tasks using large-scale training data. The first model is the predicate translation model which employs both lexical and semantic contexts to translate verbal predicates.

The second model is the argument reordering model which estimates the direction of argument movement relative to its predicate after translation. Experimental results show that both models are able to significantly improve translation accuracy in terms of BLEU score.

In the future work, we will extend our predicate translation model to translate both verbal and nominal predicates. Nominal predicates also frequently occur in Chinese sentences and thus accurate translations of them are desirable for SMT. We also want to address another translation issue of arguments as shown in Table 7: arguments are wrongly translated into separate groups instead of a cohesive unit (Wu and Fung, 2009a). We will build an argument segmentation model that follows (Xiong et al., 2011) to determine whether arguments should be translated as a unit or not.

## References

- Wilker Aziz, Miguel Rios, and Lucia Specia. 2011. Shallow semantic trees for smt. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 316–322, Edinburgh, Scotland, July. Association for Computational Linguistics.
- Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.
- Pascale Fung, Wu Zhaojun, Yang Yongsheng, and Dekai Wu. 2006. Automatic learning of chinese english semantic structure mapping. In *IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*, Aruba, December.
- Philipp Koehn, Franz Joseph Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 58–54, Edmonton, Canada, May-June.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July.
- Mamoru Komachi and Yuji Matsumoto. 2006. Phrase reordering for statistical machine translation based on predicate-argument structure. In *Proceedings of the International Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation*, pages 77–82.
- Junhui Li, Guodong Zhou, and Hwee Tou Ng. 2010. Joint syntactic and semantic parsing of chinese. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1108–1117, Uppsala, Sweden, July. Association for Computational Linguistics.
- Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 716–724, Beijing, China, August. Coling 2010 Organizing Committee.
- Arne Mauser, Saša Hasan, and Hermann Ney. 2009. Extending statistical machine translation with discriminative and trigger-based lexicon models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 210–218, Singapore, August. Association for Computational Linguistics.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia, July. Association for Computational Linguistics.
- Andreas Stolcke. 2002. Srlm—an extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, pages 901–904, Denver, Colorado, USA, September.
- Sriram Venkatapathy and Srinivas Bangalore. 2007. Three models for discriminative machine translation using global lexical selection and sentence reconstruction. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 96–102, Rochester, New York, April. Association for Computational Linguistics.
- Dekai Wu and Pascale Fung. 2009a. Can semantic role labeling improve smt. In *Proceedings of the 13th Annual Conference of the EAMT*, pages 218–225, Barcelona, May.
- Dekai Wu and Pascale Fung. 2009b. Semantic roles for smt: A hybrid two-pass model. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 13–16, Boulder, Colorado, June. Association for Computational Linguistics.
- Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Extracting pre-ordering rules from predicate-argument structures. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 29–37, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.
- Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, Australia, July. Association for Computational Linguistics.
- Deyi Xiong, Min Zhang, and Haizhou Li. 2011. A maximum-entropy segmentation model for statistical machine translation. *IEEE Transactions on Audio, Speech and Language Processing*, 19(8):2494–2505.

Nianwen Xue. 2008. Labeling chinese predicates with semantic roles. *Computational Linguistics*, 34(2):225–255.

# A Ranking-based Approach to Word Reordering for Statistical Machine Translation\*

Nan Yang<sup>†</sup>, Mu Li<sup>‡</sup>, Dongdong Zhang<sup>‡</sup>, and Nenghai Yu<sup>†</sup>

<sup>†</sup>MOE-MS Key Lab of MCC

University of Science and Technology of China

v-nayang@microsoft.com, ynh@ustc.edu.cn

<sup>‡</sup>Microsoft Research Asia

{muli, dozhang}@microsoft.com

## Abstract

Long distance word reordering is a major challenge in statistical machine translation research. Previous work has shown using source syntactic trees is an effective way to tackle this problem between two languages with substantial word order difference. In this work, we further extend this line of exploration and propose a novel but simple approach, which utilizes a ranking model based on word order precedence in the target language to reposition nodes in the syntactic parse tree of a source sentence. The ranking model is automatically derived from word aligned parallel data with a syntactic parser for source language based on both lexical and syntactical features. We evaluated our approach on large-scale Japanese-English and English-Japanese machine translation tasks, and show that it can significantly outperform the baseline phrase-based SMT system.

## 1 Introduction

Modeling word reordering between source and target sentences has been a research focus since the emerging of statistical machine translation. In phrase-based models (Och, 2002; Koehn et al., 2003), phrase is introduced to serve as the fundamental translation element and deal with local reordering, while a distance based distortion model is used to coarsely depict the exponentially decayed word movement probabilities in language translation. Further work in this direction employed lexi-

calized distortion models, including both generative (Koehn et al., 2005) and discriminative (Zens and Ney, 2006; Xiong et al., 2006) variants, to achieve finer-grained estimations, while other work took into account the hierarchical language structures in translation (Chiang, 2005; Galley and Manning, 2008).

Long-distance word reordering between language pairs with substantial word order difference, such as Japanese with Subject-Object-Verb (SOV) structure and English with Subject-Verb-Object (SVO) structure, is generally viewed beyond the scope of the phrase-based systems discussed above, because of either distortion limits or lack of discriminative features for modeling. The most notable solution to this problem is adopting syntax-based SMT models, especially methods making use of source side syntactic parse trees. There are two major categories in this line of research. One is tree-to-string model (Quirk et al., 2005; Liu et al., 2006) which directly uses source parse trees to derive a large set of translation rules and associated model parameters. The other is called *syntax pre-reordering* – an approach that re-positions source words to approximate target language word order as much as possible based on the features from source syntactic parse trees. This is usually done in a preprocessing step, and then followed by a standard phrase-based SMT system that takes the re-ordered source sentence as input to finish the translation.

In this paper, we continue this line of work and address the problem of word reordering based on source syntactic parse trees for SMT. Similar to most previous work, our approach tries to rearrange the source tree nodes sharing a common parent to mimic

\*This work has been done while the first author was visiting Microsoft Research Asia.

the word order in target language. To this end, we propose a simple but effective ranking-based approach to word reordering. The ranking model is automatically derived from the word aligned parallel data, viewing the source tree nodes to be reordered as list items to be ranked. The ranks of tree nodes are determined by their relative positions in the target language – the node in the most front gets the highest rank, while the ending word in the target sentence gets the lowest rank. The ranking model is trained to directly minimize the mis-ordering of tree nodes, which differs from the prior work based on maximum likelihood estimations of reordering patterns (Li et al., 2007; Genzel, 2010), and does not require any special tweaking in model training. The ranking model can not only be used in a pre-reordering based SMT system, but also be integrated into a phrase-based decoder serving as additional distortion features.

We evaluated our approach on large-scale Japanese-English and English-Japanese machine translation tasks, and experimental results show that our approach can bring significant improvements to the baseline phrase-based SMT system in both pre-ordering and integrated decoding settings.

In the rest of the paper, we will first formally present our ranking-based word reordering model, then followed by detailed steps of modeling training and integration into a phrase-based SMT system. Experimental results are shown in Section 5. Section 6 consists of more discussions on related work, and Section 7 concludes the paper.

## 2 Word Reordering as Syntax Tree Node Ranking

Given a source side parse tree  $T_e$ , the task of word reordering is to transform  $T_e$  to  $T'_e$ , so that  $e'$  can match the word order in target language as much as possible. In this work, we only focus on reordering that can be obtained by permuting *children* of every tree nodes in  $T_e$ . We use *children* to denote direct descendants of tree nodes for constituent trees; while for dependency trees, *children* of a node include not only all direct dependents, but also the head word itself. Figure 1 gives a simple example showing the word reordering between English and Japanese. By rearranging the position of tree nodes in the English

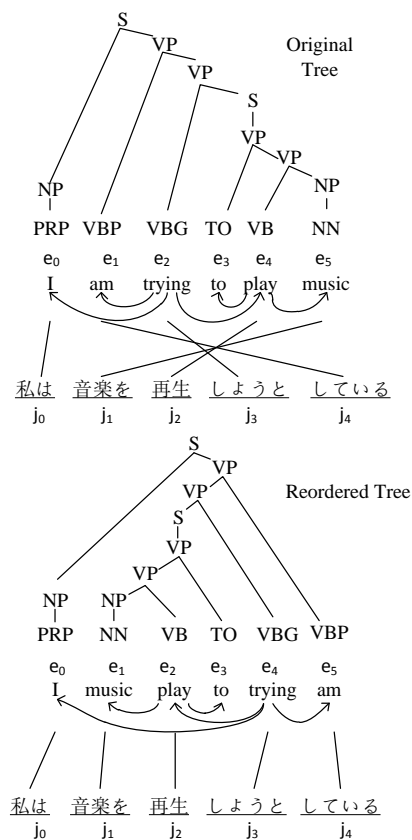


Figure 1: An English-to-Japanese sentence pair. By permuting tree nodes in the parse tree, the source sentence is reordered into the target language order. Constituent tree is shown above the source sentence; arrows below the source sentences show head-dependent arcs for dependency tree; word alignment links are lines without arrow between the source and target sentences.

parse tree, we can obtain the same word order of Japanese translation. It is true that tree-based reordering cannot cover all word movement operations in language translation, previous work showed that this method is still very effective in practice (Xu et al., 2009, Visweswariah et al., 2010).

Following this principle, the word reordering task can be broken into sub-tasks, in which we only need to determine the order of children nodes for all non-leaf nodes in the source parse tree. For a tree node  $t$  with children  $\{c_1, c_2, \dots, c_n\}$ , we rearrange the children to target-language-like order  $\{c_{\pi(i_1)}, c_{\pi(i_2)}, \dots, c_{\pi(i_n)}\}$ . If we treat the reordered position  $\pi(i)$  of child  $c_i$  as its “rank”, the reorder-

ing problem is naturally translated into a ranking problem: to reorder, we determine a “rank” for each child, then the children are sorted according to their “ranks”. As it is often impractical to directly assign a score for each permutation due to huge number of possible permutations, a widely used method is to use a real valued function  $f$  to assign a value to each node, which is called a ranking function (Herbrich et al., 2000). If we can guarantee  $(f(i) - f(j))$  and  $(\pi(i) - \pi(j))$  always has the same sign, we can get the same permutation as  $\pi$  because values of  $f$  are only used to sort the children. For example, consider the node rooted at *trying* in the dependency tree in Figure 1. Four children form a list  $\{I, am, trying, play\}$  to be ranked. Assuming ranking function  $f$  can assign values  $\{0.94, -1.83, -1.50, -1.20\}$  for  $\{I, am, trying, play\}$  respectively, we can get a sorted list  $\{I, play, trying, am\}$ , which is the desired permutation according to the target.

More formally, for a tree node  $t$  with children  $\{c_1, c_2, \dots, c_n\}$ , our ranking model assigns a rank  $f(c_i, t)$  for each child  $c_i$ , then the children are sorted according to the rank in a descending order. The ranking function  $f$  has the following form:

$$f(c_i, t) = \sum_j \theta_j(c_i, t) \cdot w_j \quad (1)$$

where the  $\theta_j$  is a feature representing the tree node  $t$  and its child  $c_i$ , and  $w_j$  is the corresponding feature weight.

### 3 Ranking Model Training

To learn ranking function in Equation (1), we need to determine the feature set  $\theta$  and learn weight vector  $w$  from reorder examples. In this section, we first describe how to extract reordering examples from parallel corpus; then we show our features for ranking function; finally, we discuss how to train the model from the extracted examples.

#### 3.1 Reorder Example Acquisition

For a sentence pair  $(e, f, a)$  with syntax tree  $T_e$  on the source side, we need to determine which reordered tree  $T'_e$  best represents the word order in target sentence  $f$ . For a tree node  $t$  in  $T_e$ , if its children align to disjoint target spans, we can simply arrange them in the order of their corresponding target

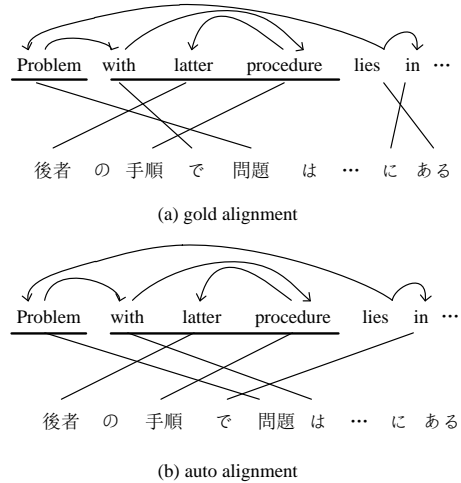


Figure 2: Fragment of a sentence pair. (a) shows gold alignment; (b) shows automatically generated alignment which contains errors.

spans. Figure 2 shows a fragment of one sentence pair in our training data. Consider the subtree rooted at word “*Problem*”. With the gold alignment, “*Problem*” is aligned to the 5th target word, and “*with latter procedure*” are aligned to target span  $[1, 3]$ , thus we can simply put “*Problem*” after “*with latter procedure*”. Recursively applying this process down the subtree, we get “*latter procedure with Problem*” which perfectly matches the target language.

As pointed out by (Li et al., 2007), in practice, nodes often have overlapping target spans due to erroneous word alignment or different syntactic structures between source and target sentences. (b) in Figure 2 shows the automatically generated alignment for the sentence pair fragment. The word “*with*” is incorrectly aligned to the 6th Japanese word “*ha*”; as a result, “*with latter procedure*” now has target span  $[1, 6]$ , while “*Problem*” aligns to  $[5, 5]$ . Due to this overlapping, it becomes unclear which permutation of “*Problem*” and “*with latter procedure*” is a better match of the target phrase; we need a better metric to measure word order similarity between reordered source and target sentences. We choose to find the tree  $T'_e$  with minimal *alignment crossing-link number (CLN)* (Genzel, 2010) to  $f$  as our golden reordered tree.<sup>1</sup> Each crossing-

<sup>1</sup>A simple solution is to exclude all trees with overlapping target spans from training. But in our experiment, this method

link  $(i_1j_1, i_2j_2)$  is a pair of alignment links crossing each other.  $CLN$  reaches zero if  $f$  is monotonically aligned to  $e'$ , and increases as there are more word reordering between  $e'$  and  $f$ . For example, in Figure 1, there are 6 crossing-links in the original tree:  $(e_1j_4, e_2j_3)$ ,  $(e_1j_4, e_4j_2)$ ,  $(e_1j_4, e_5j_1)$ ,  $(e_2j_3, e_4j_2)$ ,  $(e_2j_3, e_5j_1)$  and  $(e_4j_2, e_5j_1)$ ; thus  $CLN$  for the original tree is 6.  $CLN$  for the reordered tree is 0 as there are no crossing-links. This metric is easy to compute, and is not affected by unaligned words (Genzel, 2010).

We need to find the reordered tree with minimal  $CLN$  among all reorder candidates. As the number of candidates is in the magnitude exponential with respect to the degree of tree  $T_e$ <sup>2</sup>, it is not always computationally feasible to enumerate through all candidates. Our solution is as follows.

First, we give two definitions.

- $CLN(t)$ : the number of crossing-links  $(i_1j_1, i_2j_2)$  whose source words  $e'_{i_1}$  and  $e'_{i_2}$  both fall under sub span of the tree node  $t$ .
- $CCLN(t)$ : the number of crossing-links  $(i_1j_1, i_2j_2)$  whose source words  $e'_{i_1}$  and  $e'_{i_2}$  fall under sub span of  $t$ 's two different children nodes  $c_1$  and  $c_2$  respectively.

Apparently  $CLN$  of a tree  $T'$  equals to  $CLN(\text{root of } T')$ , and  $CLN(t)$  can be recursively expressed as:

$$CLN(t) = CCLN(t) + \sum_{\text{child } c \text{ of } t} CLN(c)$$

Take the original tree in Figure 1 for example. At the root node *trying*,  $CLN(\text{trying})$  is 6 because there are six crossing-links under its sub-span:  $(e_1j_4, e_2j_3)$ ,  $(e_1j_4, e_4j_2)$ ,  $(e_1j_4, e_5j_1)$ ,  $(e_2j_3, e_4j_2)$ ,  $(e_2j_3, e_5j_1)$  and  $(e_4j_2, e_5j_1)$ . On the other hand,  $CCLN(\text{trying})$  is 5 because  $(e_4j_2, e_5j_1)$  falls under its child node *play*, thus does not count towards  $CCLN$  of *trying*.

From the definition, we can easily see that  $CCLN(t)$  can be determined solely by the order of  $t$ 's direct children, and  $CLN(t)$  is only affected by

<sup>2</sup>discarded too many training instances and led to degraded reordering performance.

<sup>2</sup>In our experiments, there are nodes with more than 10 children for English dependency trees.

the reorder in the subtree of  $t$ . This observation enables us to divide the task of finding the reordered tree  $T'_e$  with minimal  $CLN$  into independently finding the children permutation of each node with minimal  $CCLN$ . Unfortunately, the time cost for the sub-task is still  $O(n!)$  for a node with  $n$  children. Instead of enumerating through all permutations, we only search the *Inversion Transduction Grammar neighborhood* of the initial sequence (Tromble, 2009). As pointed out by (Tromble, 2009), the ITG neighborhood is large enough for reordering task, and can be searched through efficiently using a CKY decoder.

After finding the best reordered tree  $T'_e$ , we can extract one reorder example from every node with more than one child.

### 3.2 Features

Features for the ranking model are extracted from source syntax trees. For English-to-Japanese task, we extract features from Stanford English Dependency Tree (Marneffe et al., 2006), including lexicons, Part-of-Speech tags, dependency labels, punctuations and tree distance between head and dependent. For Japanese-to-English task, we use a chunk-based Japanese dependency tree (Kudo and Matsumoto, 2002). Different from features for English, we do not use dependency labels because they are not available from the Japanese parser. Additionally, Japanese function words are also included as features because they are important grammatical clues. The detailed feature templates are shown in Table 1.

### 3.3 Learning Method

There are many well studied methods available to learn the ranking function from extracted examples., ListNet (?) etc. We choose to use RankingSVM (Herbrich et al., 2000), a pair-wised ranking method, for its simplicity and good performance.

For every reorder example  $t$  with children  $\{c_1, c_2, \dots, c_n\}$  and their desired permutation  $\{c_{\pi(i_1)}, c_{\pi(i_2)}, \dots, c_{\pi(i_n)}\}$ , we decompose it into a set of pair-wised training instances. For any two children nodes  $c_i$  and  $c_j$  with  $i < j$ , we extract a positive instance if  $\pi(i) < \pi(j)$ , otherwise we extract a negative instance. The feature vector for both positive instance and negative instance is  $(\theta_{c_i} - \theta_{c_j})$ , where  $\theta_{c_i}$  and  $\theta_{c_j}$  are feature vectors for  $c_i$  and  $c_j$

E-J		
$c_l$	$c_l \cdot dst$	$c_l \cdot pct$
$c_l \cdot dst \cdot pct$	$c_l \cdot lc_l$	$c_l \cdot rc_l$
$c_l \cdot lc_l \cdot dst$	$c_l \cdot rc_l \cdot dst$	$c_l \cdot c_{lex}$
$c_l \cdot c_{lex}$	$c_l \cdot c_{lex} \cdot dst$	$c_l \cdot c_{lex} \cdot dst$
$c_l \cdot h_{lex}$	$c_l \cdot h_{lex}$	$c_l \cdot h_{lex} \cdot dst$
$c_l \cdot h_{lex} \cdot dst$	$c_l \cdot c_{lex} \cdot pct$	$c_l \cdot c_{lex} \cdot pct$
$c_l \cdot h_{lex} \cdot pct$	$c_l \cdot h_{lex} \cdot pct$	
J-E		
$c_{tf}$	$c_{tf} \cdot dst$	$c_{tf} \cdot lc_t$
$c_{tf} \cdot rc_t$	$c_{tf} \cdot lc_t \cdot dst$	$c_l \cdot rc_t \cdot dst$
$c_{tf} \cdot c_{lex}$	$c_{tf} \cdot c_{lex}$	$c_{tf} \cdot c_{lex} \cdot dst$
$c_{tf} \cdot c_{lex} \cdot dst$	$c_{tf} \cdot h_f$	$c_{tf} \cdot h_f$
$c_{tf} \cdot h_f \cdot dst$	$c_{tf} \cdot h_f \cdot dst$	$c_{tf} \cdot h_{lex}$
$c_{tf} \cdot h_{lex}$	$c_{tf} \cdot h_{lex} \cdot dst$	$c_{tf} \cdot h_{lex} \cdot dst$

Table 1: Feature templates for ranking function. All templates are implicitly conjuncted with the pos tag of head node.

$c$ : child to be ranked;  $h$ : head node  
 $lc$ : left sibling of  $c$ ;  $rc$ : right sibling of  $c$   
 $l$ : dependency label;  $t$ : pos tag  
 $lex$ : top frequency lexicons  
 $f$ : Japanese function word  
 $dst$ : tree distance between  $c$  and  $h$   
 $pct$ : punctuation node between  $c$  and  $h$

respectively. In this way, ranking function learning is turned into a simple binary classification problem, which can be easily solved by a two-class linear support vector machine.

## 4 Integration into SMT system

There are two ways to integrate the ranking reordering model into a phrase-based SMT system: the pre-reorder method, and the decoding time constraint method.

For pre-reorder method, ranking reorder model is applied to reorder source sentences during both training and decoding. Reordered sentences can go through the normal pipeline of a phrase-based decoder.

The ranking reorder model can also be integrated into a phrase based decoder. Integrated method takes the original source sentence  $e$  as input, and ranking model generates a reordered  $e'$  as a word order ref-

erence for the decoder. A simple penalty scheme is utilized to penalize decoder reordering violating ranking reorder model's prediction  $e'$ . In this paper, our underlying decoder is a CKY decoder following Bracketing Transduction Grammar (Wu, 1997; Xiong et al., 2006), thus we show how the penalty is implemented in the BTG decoder as an example. Similar penalty can be designed for other decoders without much effort.

Under BTG, three rules are used to derive translations: one unary terminal rule, one *straight* rule and one *inverse* rule:

$$\begin{aligned}
 A &\rightarrow e/f \\
 A &\rightarrow [A_1, A_2] \\
 A &\rightarrow \langle A_1, A_2 \rangle
 \end{aligned}$$

We have three penalty triggers when any rules are applied during decoding:

- Discontinuous penalty  $f_{dc}$ : it fires for all rules when source span of either  $A$ ,  $A_1$  or  $A_2$  is mapped to discontinuous span in  $e'$ .
- Wrong straight rule penalty  $f_{st}$ : it fires for straight rule when source spans of  $A_1$  and  $A_2$  are not mapped to two adjacent spans in  $e'$  in straight order.
- Wrong inverse rule penalty  $f_{iv}$ : it fires for inverse rule when source spans of  $A_1$  and  $A_2$  are not mapped to two adjacent spans in  $e'$  in inverse order.

The above three penalties are added as additional features into the log-linear model of the phrase-based system. Essentially they are soft constraints to encourage the decoder to choose translations with word order similar to the prediction of ranking reorder model.

## 5 Experiments

To test our ranking reorder model, we carry out experiments on large scale English-To-Japanese, and Japanese-To-English translation tasks.

### 5.1 Data

#### 5.1.1 Evaluation Data

We collect 3,500 Japanese sentences and 3,500 English sentences from the web. They come from



a wide range of domains, such as technical documents, web forum data, travel logs etc. They are manually translated into the other language to produce 7,000 sentence pairs, which are split into two parts: 2,000 pairs as development set (*dev*) and the other 5,000 pairs as test set (*web* test).

Beside that, we collect another 999 English sentences from newswire domain which are translated into Japanese to form an out-of-domain test data set (*news* test).

### 5.1.2 Parallel Corpus

Our parallel corpus is crawled from the web, containing news articles, technical documents, blog entries etc. After removing duplicates, we have about 18 million sentence pairs, which contain about 270 millions of English tokens and 320 millions of Japanese tokens. We use Giza++ (Och and Ney, 2003) to generate the word alignment for the parallel corpus.

### 5.1.3 Monolingual Corpus

Our monolingual Corpus is also crawled from the web. After removing duplicate sentences, we have a corpus of over 10 billion tokens for both English and Japanese. This monolingual corpus is used to train a 4-gram language model for English and Japanese respectively.

## 5.2 Parsers

For English, we train a dependency parser as (Nivre and Scholz, 2004) on WSJ portion of Penn Treebank, which are converted to dependency trees using Stanford Parser (Marneffe et al., 2006). We convert the tokens in training data to lower case, and re-tokenize the sentences using the same tokenizer from our MT system.

For Japanese parser, we use CABOCHA, a chunk-based dependency parser (Kudo and Matsumoto, 2002). Some heuristics are used to adapt CABOCHA generated trees to our word segmentation.

## 5.3 Settings

### 5.3.1 Baseline System

We use a BTG phrase-based system with a Max-Ent based lexicalized reordering model (Wu, 1997; Xiong et al., 2006) as our baseline system for

both English-to-Japanese and Japanese-to-English Experiment. The distortion model is trained on the same parallel corpus as the phrase table using a home implemented maximum entropy trainer.

In addition, a pre-reorder system using manual rules as (Xu et al., 2009) is included for the English-to-Japanese experiment (ManR-PR). Manual rules are tuned by a bilingual speaker on the development set.

### 5.3.2 Ranking Reordering System

Ranking reordering model is learned from the same parallel corpus as phrase table. For efficiency reason, we only use 25% of the corpus to train our reordering model. LIBLINEAR (Fan et al., 2008) is used to do the SVM optimization for RankingSVM.

We test it on both pre-reorder setting (*Rank-PR*) and integrated setting (*Rank-IT*).

## 5.4 End-to-End Result

	system	dev	web test	news test
E-J	Baseline	21.45	21.12	14.18
	ManR-PR	23.00	22.42	15.61
	Rank-PR	22.92	22.51	<b>15.90</b>
	Rank-IT	<b>23.14</b>	<b>22.85</b>	15.72
J-E	Baseline	25.39	24.20	14.26
	Rank-PR	26.57	25.56	<b>15.42</b>
	Rank-IT	<b>26.72</b>	<b>25.87</b>	15.27

Table 2: BLEU(%) score on dev and test data for both E-J and J-E experiment. All settings significantly improve over the baseline at 95% confidence level. Baseline is the BTG phrase system system; ManR-PR is pre-reorder with manual rule; Rank-PR is pre-reorder with ranking reorder model; Rank-IT is system with integrated ranking reorder model.

From Table 2, we can see our ranking reordering model significantly improves the performance for both English-to-Japanese and Japanese-to-English experiments over the BTG baseline system. It also out-performs the manual rule set on English-to-Japanese result, but the difference is not significant.

## 5.5 Reordering Performance

In order to show whether the improved performance is really due to improved reordering, we would like to measure the reorder performance directly.

As we do not have access to a golden re-ordered sentence set, we decide to use the alignment crossing-link numbers between aligned sentence pairs as the measure for reorder performance.

We train the ranking model on 25% of our parallel corpus, and use the rest 75% as test data (*auto*). We sample a small corpus (575 sentence pairs) and do manual alignment (*man-small*). We denote the automatic alignment for these 575 sentences as (*auto-small*). From Table 3, we can see

	setting	auto	auto-small	man-small
	None	36.3	35.9	40.1
E-J	Oracle	4.3	4.1	7.4
	ManR	13.4	13.6	16.7
	Rank	12.1	12.8	17.2
J-E	Oracle	6.9	7.0	9.4
	Rank	15.7	15.3	20.5

Table 3: Reorder performance measured by crossing-link number per sentence. *None* means the original sentences without reordering; *Oracle* means the best permutation allowed by the source parse tree; *ManR* refers to manual reorder rules; *Rank* means ranking reordering model.

our ranking reordering model indeed significantly reduces the crossing-link numbers over the original sentence pairs. On the other hand, the performance of the ranking reorder model still fall far short of oracle, which is the lowest crossing-link number of all possible permutations allowed by the parse tree. By manual analysis, we find that the gap is due to both errors of the ranking reorder model and errors from word alignment and parser.

Another thing to note is that the crossing-link number of manual alignment is higher than automatic alignment. The reason is that our annotators tend to align function words which might be left unaligned by automatic word aligner.

### 5.6 Effect of Ranking Features

Here we examine the effect of features for ranking reorder model. We compare their influence on RankingSVM accuracy, alignment crossing-link number, end-to-end BLEU score, and the model size. As Table 4 shows, a major part of reduction of *CLN* comes from features such as Part-of-Speech tags,

	Features	Acc.	<i>CLN</i>	BLEU	Feat.#
E-J	<i>tag+label</i>	88.6	16.4	22.24	26k
	<i>+dst</i>	91.5	13.5	22.66	55k
	<i>+pct</i>	92.2	13.1	22.73	79k
	<i>+lex<sub>100</sub></i>	92.9	12.1	22.85	347k
	<i>+lex<sub>1000</sub></i>	94.0	11.5	22.79	2,410k
	<i>+lex<sub>2000</sub></i>	95.2	10.7	22.81	3,794k
J-E	<i>tag+fw</i>	85.0	18.6	25.43	31k
	<i>+dst</i>	90.3	16.9	25.62	65k
	<i>+lex<sub>100</sub></i>	91.6	15.7	25.87	293k
	<i>+lex<sub>1000</sub></i>	92.4	14.8	25.91	2,156k
	<i>+lex<sub>2000</sub></i>	93.0	14.3	25.84	3,297k

Table 4: Effect of ranking features. Acc. is RankingSVM accuracy in percentage on the training data; *CLN* is the crossing-link number per sentence on parallel corpus with automatically generated word alignment; BLEU is the BLEU score in percentage on **web** test set on *Rank-IT* setting (system with integrated rank reordering model); *lex<sub>n</sub>* means n most frequent lexicons in the training corpus.

dependency labels (for English), function words (for Japanese), and the distance and punctuations between child and head. These features also correspond to BLEU score improvement for End-to-End evaluations. Lexicon features generally continue to improve the RankingSVM accuracy and reduce *CLN* on training data, but they do not bring further improvement for SMT systems beyond the top 100 most frequent words. Our explanation is that less frequent lexicons tend to help local reordering only, which is already handled by the underlying phrase-based system.

### 5.7 Performance on different domains

From Table 2 we can see that pre-reorder method has higher BLEU score on **news** test, while integrated model performs better on **web** test set which contains informal texts. By error analysis, we find that the parser commits more errors on informal texts, and informal texts usually have more flexible translations. Pre-reorder method makes “hard” decision before decoding, thus is more sensitive to parser errors; on the other hand, integrated model is forced to use a longer distortion limit which leads to more search errors during decoding time. It is possible to

use system combination method to get the best of both systems, but we leave this to future work.

## 6 Discussion on Related Work

There have been several studies focusing on compiling hand-crafted syntactic reorder rules. Collins et al. (2005), Wang et al. (2007), Ramanathan et al. (2008), Lee et al. (2010) have developed rules for German-English, Chinese-English, English-Hindi and English-Japanese respectively. Xu et al. (2009) designed a clever precedence reordering rule set for translation from English to several SOV languages. The drawback for hand-crafted rules is that they depend upon expert knowledge to produce and are limited to their targeted language pairs.

Automatically learning syntactic reordering rules have also been explored in several work. Li et al. (2007) and Visweswariah et al. (2010) learned probability of reordering patterns from constituent trees using either Maximum Entropy or maximum likelihood estimation. Since reordering patterns are matched against a tree node together with all its direct children, data sparseness problem will arise when tree nodes have many children (Li et al., 2007); Visweswariah et al. (2010) also mentioned their method yielded no improvement when applied to dependency trees in their initial experiments. Genzel (2010) dealt with the data sparseness problem by using window heuristic, and learned reordering pattern sequence from dependency trees. Even with the window heuristic, they were unable to evaluate all candidates due to the huge number of possible patterns. Different from the previous approaches, we treat syntax-based reordering as a ranking problem between different source tree nodes. Our method does not require the source nodes to match some specific patterns, but encodes reordering knowledge in the form of a ranking function, which naturally handles reordering between any number of tree nodes; the ranking function is trained by well-established rank learning method to minimize the number of mis-ordered tree nodes in the training data.

Tree-to-string systems (Quirk et al., 2005; Liu et al., 2006) model syntactic reordering using minimal or composed translation rules, which may contain reordering involving tree nodes from multiple tree

levels. Our method can be naturally extended to deal with such multiple level reordering. For a tree-to-string rule with multiple tree levels, instead of ranking the direct children of the root node, we rank all leaf nodes (Most are frontier nodes (Galley et al., 2006)) in the translation rule. We need to redesign our ranking feature templates to encode the reordering information in the source part of the translation rules. We need to remember the source side context of the rules, the model size would still be much smaller than a full-fledged tree-to-string system because we do not need to explicitly store the target variants for each rule.

## 7 Conclusion and Future Work

In this paper we present a ranking based reordering method to reorder source language to match the word order of target language given the source side parse tree. Reordering is formulated as a task to rank different nodes in the source side syntax tree according to their relative position in the target language. The ranking model is automatically trained to minimize the mis-ordering of tree nodes in the training data. Large scale experiment shows improvement on both reordering metric and SMT performance, with up to 1.73 point BLEU gain in our evaluation test.

In future work, we plan to extend the ranking model to handle reordering between multiple levels of source trees. We also expect to explore better way to integrate ranking reorder model into SMT system instead of a simple penalty scheme. Along the research direction of preprocessing the source language to facilitate translation, we consider to not only change the order of the source language, but also inject syntactic structure of the target language into source language by adding pseudo words into source sentences.

## Acknowledgements

Nan Yang and Nenghai Yu were partially supported by Fundamental Research Funds for the Central Universities (No. WK2100230002), National Natural Science Foundation of China (No. 60933013), and National Science and Technology Major Project (No. 2010ZX03004-003).

## References

- David Chiang. 2005. *A Hierarchical Phrase-Based Model for Statistical Machine Translation*. In *Proc. ACL*, pages 263-270.
- Michael Collins, Philipp Koehn and Ivona Kucerova. 2005. *Clause restructuring for statistical machine translation*. In *Proc. ACL*.
- R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. 2008. *LIBLINEAR: A library for large linear classification*. In *Journal of Machine Learning Research*.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. *Scalable Inference and Training of Context-Rich Syntactic Translation Models*. In *Proc. ACL-Coling*, pages 961-968.
- Michel Galley and Christopher D. Manning. 2008. *A Simple and Effective Hierarchical Phrase Reordering Model*. In *Proc. EMNLP*, pages 263-270.
- Dmitriy Genzel. 2010. *Automatically Learning Source-side Reordering Rules for Large Scale Machine Translation*. In *Proc. Coling*, pages 376-384.
- Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 2000. *Large Margin Rank Boundaries for Ordinal Regression*. In *Advances in Large Margin Classifiers*, pages 115-132.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne and David Talbot. 2005. *Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation*. In *International Workshop on Spoken Language Translation*.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. *Statistical Phrase-Based Translation*. In *Proc. HLT-NAACL*, pages 127-133.
- Taku Kudo, Yuji Matsumoto. 2002. *Japanese Dependency Analysis using Cascaded Chunking*. In *Proc. CoNLL*, pages 63-69.
- Young-Suk Lee, Bing Zhao and Xiaoqiang Luo. 2010. *Constituent reordering and syntax models for English-to-Japanese statistical machine translation*. In *Proc. Coling*.
- Chi-Ho Li, Minghui Li, Dongdong Zhang, Mu Li and Ming Zhou and Yi Guan. 2007. *A Probabilistic Approach to Syntax-based Reordering for Statistical Machine Translation*. In *Proc. ACL*, pages 720-727.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. *Tree-to-String Alignment Template for Statistical Machine Translation*. In *Proc. ACL-Coling*, pages 609-616.
- Marie-Catherine de Marneffe, Bill MacCartney and Christopher D. Manning. 2006. *Generating Typed Dependency Parses from Phrase Structure Parses*. In *LREC 2006*
- Joakim Nivre and Mario Scholz. 2004. *Deterministic Dependency Parsing for English Text*. In *Proc. Coling*.
- Franz J. Och. 2002. *Statistical Machine Translation: From Single Word Models to Alignment Template*. Ph.D. Thesis, RWTH Aachen, Germany
- Franz J. Och and Hermann Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models*. *Computational Linguistics*, 29(1): pages 19-51.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. *Dependency Treelet Translation: Syntactically Informed Phrasal SMT*. In *Proc. ACL*, pages 271-279.
- A. Ramanathan, Pushpak Bhattacharyya, Jayprasad Hegde, Ritesh M. Shah and Sasikumar M. 2008. *Simple syntactic and morphological processing can help English-Hindi Statistical Machine Translation*. In *Proc. IJCNLP*.
- Roy Tromble. 2009. *Search and Learning for the Linear Ordering Problem with an Application to Machine Translation*. Ph.D. Thesis.
- Karthik Visweswariah, Jiri Navratil, Jeffrey Sorensen, Vijil Chenthamarakshan and Nandakishore Kambhatla. 2010. *Syntax Based Reordering with Automatically Derived Rules for Improved Statistical Machine Translation*. In *Proc. Coling*, pages 1119-1127.
- Chao Wang, Michael Collins, Philipp Koehn. 2007. *Chinese syntactic reordering for statistical machine translation*. In *Proc. EMNLP-CoNLL*.
- Dekai Wu. 1997. *Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora*. *Computational Linguistics*, 23(3): pages 377-403.
- Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. *Maximum Entropy Based Phrase Reordering Model for Statistical Machine Translation*. In *Proc. ACL-Coling*, pages 521-528.
- Peng Xu, Jaeho Kang, Michael Ringgaard, Franz Och. 2009. *Using a Dependency Parser to Improve SMT for Subject-Object-Verb Languages*. In *Proc. HLT-NAACL*, pages 376-384.
- Richard Zens and Hermann Ney. 2006. *Discriminative Reordering Models for Statistical Machine Translation*. In *Proc. Workshop on Statistical Machine Translation, HLT-NAACL*, pages 127-133.

# Character-Level Machine Translation Evaluation for Languages with Ambiguous Word Boundaries

Chang Liu and Hwee Tou Ng  
Department of Computer Science  
National University of Singapore  
13 Computing Drive, Singapore 117417  
{liuchan1, nght}@comp.nus.edu.sg

## Abstract

In this work, we introduce the TESLA-CELAB metric (Translation Evaluation of Sentences with Linear-programming-based Analysis – Character-level Evaluation for Languages with Ambiguous word Boundaries) for automatic machine translation evaluation. For languages such as Chinese where words usually have meaningful internal structure and word boundaries are often fuzzy, TESLA-CELAB acknowledges the advantage of character-level evaluation over word-level evaluation. By reformulating the problem in the linear programming framework, TESLA-CELAB addresses several drawbacks of the character-level metrics, in particular the modeling of synonyms spanning multiple characters. We show empirically that TESLA-CELAB significantly outperforms character-level BLEU in the English-Chinese translation evaluation tasks.

## 1 Introduction

Since the introduction of BLEU (Papineni et al., 2002), automatic machine translation (MT) evaluation has received a lot of research interest. The Workshop on Statistical Machine Translation (WMT) hosts regular campaigns comparing different machine translation evaluation metrics (Callison-Burch et al., 2009; Callison-Burch et al., 2010; Callison-Burch et al., 2011). In the WMT shared tasks, many new generation metrics, such as METEOR (Banerjee and Lavie, 2005), TER (Snover et al., 2006), and TESLA (Liu et al., 2010) have consistently outperformed BLEU as judged by the correlations with human judgments.

The research on automatic machine translation evaluation is important for a number of reasons. Automatic translation evaluation gives machine translation researchers a cheap and reproducible way to guide their research and makes it possible to compare machine translation methods across different studies. In addition, machine translation system parameters are tuned by maximizing the automatic scores. Some recent research (Liu et al., 2011) has shown evidence that replacing BLEU by a newer metric, TESLA, can improve the human judged translation quality.

Despite the importance and the research interest on automatic MT evaluation, almost all existing work has focused on European languages, in particular on English. Although many methods aim to be language neutral, languages with very different characteristics such as Chinese do present additional challenges. The most obvious challenge for Chinese is that of word segmentation.

Unlike European languages, written Chinese is not split into words. Segmenting Chinese sentences into words is a natural language processing task in its own right (Zhao and Liu, 2010; Low et al., 2005). However, many different segmentation standards exist for different purposes, such as Microsoft Research Asia (MSRA) for Named Entity Recognition (NER), Chinese Treebank (CTB) for parsing and part-of-speech (POS) tagging, and City University of Hong Kong (CITYU) and Academia Sinica (AS) for general word segmentation and POS tagging. It is not clear which standard is the best in a given scenario.

The only prior work attempting to address the problem of word segmentation in automatic MT evaluation for Chinese that we are aware of is Li et

买	伞	
buy	umbrella	
买	雨伞	
buy	umbrella	
买	雨	伞
buy	rain	umbrella

Figure 1: Three forms of the same expression *buy umbrella* in Chinese

al. (2011). The work compared various MT evaluation metrics (BLEU, NIST, METEOR, GTM, 1 – TER) with different segmentation schemes, and found that treating every single character as a token (character-level MT evaluation) gives the best correlation with human judgments.

## 2 Motivation

Li et al. (2011) identify two reasons that character-based metrics outperform word-based metrics. For illustrative purposes, we use Figure 1 as a running example in this paper. All three expressions are semantically identical (*buy umbrella*). The first two forms are identical because 雨伞<sup>1</sup> and 伞 are synonyms. The last form is simply an (arguably wrong) alternative segmented form of the second expression.

1. Word-based metrics do not award partial matches, e.g., 买\_雨伞 and 买\_伞 would be penalized for the mismatch between 雨伞 and 伞. Character-based metrics award the match between characters 伞 and 伞.
2. Character-based metrics do not suffer from errors and differences in word segmentation, so 买\_雨伞 and 买\_雨\_伞 would be judged exactly equal.

Li et al. (2011) conduct empirical experiments to show that character-based metrics consistently outperform their word-based counterparts. Despite that, we observe two important problems for the character-based metrics:

1. Although partial matches are partially awarded, the mechanism breaks down for n-grams where

<sup>1</sup>Literally, *rain umbrella*.

$n > 1$ . For example, between 买\_雨\_伞 and 买\_伞, higher-order n-grams such as 买\_雨 and 雨\_伞 still have no match, and will be penalized accordingly, even though 买\_雨\_伞 and 买\_伞 should match exactly. N-grams such as 买\_雨 which cross natural word boundaries and are meaningless by themselves can be particularly tricky.

2. Character-level metrics can utilize only a small part of the Chinese synonym dictionary, such as 你 and 您 (*you*). The majority of Chinese synonyms involve more than one character, such as 雨伞 and 伞 (*umbrella*), and 儿童 and 小孩 (*child*).

In this work, we attempt to address both of these issues by introducing TESLA-CELAB, a character-level metric that also models word-level linguistic phenomenon. We formulate the n-gram matching process as a real-valued linear programming problem, which can be solved efficiently. The metric is based on the TESLA automatic MT evaluation framework (Liu et al., 2010; Dahlmeier et al., 2011).

## 3 The Algorithm

### 3.1 Basic Matching

We illustrate our matching algorithm using the examples in Figure 1. Let 买雨伞 be the reference, and 买伞 be the candidate translation.

We use Cilin (同义词词林)<sup>2</sup> as our synonym dictionary. The basic n-gram matching problem is shown in Figure 2. Two n-grams are connected if they are identical, or if they are identified as synonyms by Cilin. Notice that all n-grams are put in the same matching problem regardless of  $n$ , unlike in translation evaluation metrics designed for European languages. This enables us to designate n-grams with different values of  $n$  as synonyms, such as 雨伞 ( $n = 2$ ) and 伞 ( $n = 1$ ).

In this example, we are able to make a total of two successful matches. The recall is therefore 2/6 and the precision is 2/3.

<sup>2</sup>[http://ir.hit.edu.cn/phpwebsite/index.php?module=pagemaster&PAGE\\_user\\_op=view\\_page&PAGE\\_id=162](http://ir.hit.edu.cn/phpwebsite/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=162)

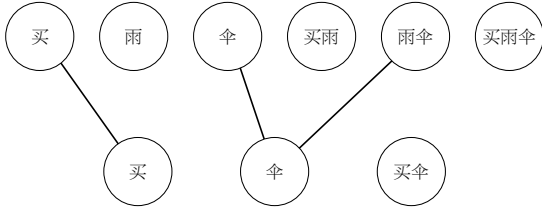


Figure 2: The basic n-gram matching problem

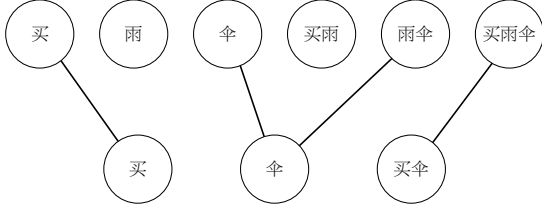


Figure 3: The n-gram matching problem after phrase matching

### 3.2 Phrase Matching

We note in Figure 2 that the trigram 买雨伞 and the bigram 买伞 are still unmatched, even though the match between 雨伞 and 伞 should imply the match between 买雨伞 and 买伞.

We infer the matching of such phrases using a dynamic programming algorithm. Two n-grams are considered synonyms if they can be segmented into synonyms that are aligned. With this extension, we are able to match 买雨伞 and 买伞 (since 买 matches 买 and 雨伞 matches 伞). The matching problem is now depicted by Figure 3.

The linear programming problem is mathematically described as follows. The variables  $w(\cdot, \cdot)$  are the weights assigned to the edges,

$$\begin{aligned} w(\text{买}, \text{买}) &\in [0, 1] \\ w(\text{伞}, \text{伞}) &\in [0, 1] \\ w(\text{雨伞}, \text{伞}) &\in [0, 1] \\ w(\text{买雨伞}, \text{买伞}) &\in [0, 1] \end{aligned}$$

We require that for any node  $N$ , the sum of weights assigned to edges linking  $N$  must not exceed one.

$$\begin{aligned} w_{\text{ref}}(\text{买}) &= w(\text{买}, \text{买}) \\ w_{\text{ref}}(\text{伞}) &= w(\text{伞}, \text{伞}) \\ w_{\text{ref}}(\text{雨伞}) &= w(\text{雨伞}, \text{伞}) \\ w_{\text{ref}}(\text{买雨伞}) &= w(\text{买雨伞}, \text{买伞}) \end{aligned}$$

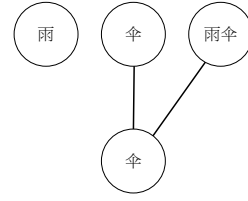


Figure 4: A covered n-gram matching problem

$$\begin{aligned} w_{\text{cand}}(\text{买}) &= w(\text{买}, \text{买}) \\ w_{\text{cand}}(\text{伞}) &= w(\text{伞}, \text{伞}) + w(\text{雨伞}, \text{伞}) \\ w_{\text{cand}}(\text{买伞}) &= w(\text{买雨伞}, \text{买伞}) \end{aligned}$$

where

$$\begin{aligned} w_{\text{ref}}(X) &\in [0, 1] && \forall X \\ w_{\text{cand}}(X) &\in [0, 1] && \forall X \end{aligned}$$

Now we maximize the total match,

$$w(\text{买}, \text{买}) + w(\text{伞}, \text{伞}) + w(\text{雨伞}, \text{伞}) + w(\text{买雨伞}, \text{买伞})$$

In this example, the best match is 3, resulting in a recall of 3/6 and a precision of 3/3.

### 3.3 Covered Matching

In Figure 3, n-grams 雨 and 买雨 in the reference remain impossible to match, which implies misguided penalty for the candidate translation. We observe that since 买雨伞 has been matched, all its sub-n-grams should be considered matched as well, including 雨 and 买雨. We call this the covered n-gram matching rule. This relationship is implicit in the matching problem for English translation evaluation metrics where words are well delimited. But with phrase matching in Chinese, it must be modeled explicitly.

However, we cannot simply perform covered n-gram matching as a post processing step. As an example, suppose we are matching phrases 雨伞 and 伞, as shown in Figure 4. The linear programming solver may come up with any of the solutions where  $w(\text{伞}, \text{伞}) + w(\text{雨伞}, \text{伞}) = 1$ ,  $w(\text{伞}, \text{伞}) \in [0, 1]$ , and  $w(\text{雨伞}, \text{伞}) \in [0, 1]$ .

To give the maximum coverage for the node 雨, only the solution  $w(\text{伞}, \text{伞}) = 0$ ,  $w(\text{雨伞}, \text{伞}) = 1$  is accepted. This indicates the need to model covered

n-gram matching in the linear programming problem itself.

We return to the matching of the reference 买雨伞 and the candidate 买伞 in Figure 3. On top of the  $w(\cdot)$  variables already introduced, we add the variables *maximum covering weights*  $c(\cdot)$ . Each  $c(X)$  represents the maximum  $w(Y)$  variable where n-gram  $Y$  completely covers n-gram  $X$ .

$$\begin{aligned}
c_{\text{ref}}(\text{买}) &\leq \max(w_{\text{ref}}(\text{买}), w_{\text{ref}}(\text{买雨}), \\
&\quad w_{\text{ref}}(\text{买雨伞})) \\
c_{\text{ref}}(\text{雨}) &\leq \max(w_{\text{ref}}(\text{雨}), w_{\text{ref}}(\text{买雨}), \\
&\quad w_{\text{ref}}(\text{雨伞}), w_{\text{ref}}(\text{买雨伞})) \\
c_{\text{ref}}(\text{伞}) &\leq \max(w_{\text{ref}}(\text{伞}), w_{\text{ref}}(\text{雨伞}), \\
&\quad w_{\text{ref}}(\text{买雨伞})) \\
c_{\text{ref}}(\text{买雨}) &\leq \max(w_{\text{ref}}(\text{买雨}), w_{\text{ref}}(\text{买雨伞})) \\
c_{\text{ref}}(\text{雨伞}) &\leq \max(w_{\text{ref}}(\text{雨伞}), w_{\text{ref}}(\text{买雨伞})) \\
c_{\text{ref}}(\text{买雨伞}) &\leq w_{\text{ref}}(\text{买雨伞}) \\
c_{\text{cand}}(\text{买}) &\leq \max(w_{\text{cand}}(\text{买}), w_{\text{cand}}(\text{买伞})) \\
c_{\text{cand}}(\text{伞}) &\leq \max(w_{\text{cand}}(\text{伞}), w_{\text{cand}}(\text{买伞})) \\
c_{\text{cand}}(\text{买伞}) &\leq w_{\text{cand}}(\text{买伞})
\end{aligned}$$

where

$$\begin{aligned}
c_{\text{ref}}(X) &\in [0, 1] && \forall X \\
c_{\text{cand}}(X) &\in [0, 1] && \forall X
\end{aligned}$$

However, the  $\max(\cdot)$  operator is not allowed in the linear programming formulation. We get around this by approximating  $\max(\cdot)$  with the sum instead. Hence,

$$\begin{aligned}
c_{\text{ref}}(\text{买}) &\leq w_{\text{ref}}(\text{买}) + w_{\text{ref}}(\text{买雨}) + \\
&\quad w_{\text{ref}}(\text{买雨伞}) \\
c_{\text{ref}}(\text{雨}) &\leq w_{\text{ref}}(\text{雨}) + w_{\text{ref}}(\text{买雨}) + \\
&\quad w_{\text{ref}}(\text{雨伞}) + w_{\text{ref}}(\text{买雨伞}) \\
&\dots
\end{aligned}$$

We justify this approximation by the following observation. Consider the sub-problem consisting of just the  $w(\cdot, \cdot)$ ,  $w_{\text{ref}}(\cdot)$ ,  $w_{\text{cand}}(\cdot)$  variables and their associated constraints. This sub-problem can be seen as a maximum flow problem where all constants are integers, hence there exists an optimal solution where each of the  $w$  variables is assigned a value of either 0 or 1. For such a solution, the

$\max$  and the sum forms are equivalent, since the  $c_{\text{ref}}(\cdot)$  and  $c_{\text{cand}}(\cdot)$  variables are also constrained to the range  $[0, 1]$ .

The maximum flow equivalence breaks down when the  $c(\cdot)$  variables are introduced, so in the general case, replacing  $\max$  with sum is only an approximation.

Returning to our sample problem, the linear programming solver simply needs to assign:

$$\begin{aligned}
w(\text{买雨伞}, \text{买伞}) &= 1 \\
w_{\text{ref}}(\text{买雨伞}) &= 1 \\
w_{\text{cand}}(\text{买伞}) &= 1
\end{aligned}$$

Consequently, due to the maximum covering weights constraint, we can give the following value assignment, implying that all n-grams have been matched.

$$\begin{aligned}
c_{\text{ref}}(X) &= 1 && \forall X \\
c_{\text{cand}}(X) &= 1 && \forall X
\end{aligned}$$

### 3.4 The Objective Function

We now define our objective function in terms of the  $c(\cdot)$  variables. The recall is a function of  $\sum_X c_{\text{ref}}(X)$ , and the precision is a function of  $\sum_Y c_{\text{cand}}(Y)$ , where  $X$  is the set of all n-grams of the reference, and  $Y$  is the set of all n-grams of the candidate translation.

Many prior translation evaluation metrics such as MAXSIM (Chan and Ng, 2008) and TESLA (Liu et al., 2010; Dahlmeier et al., 2011) use the F-0.8 measure as the final score:

$$F_{0.8} = \frac{\text{Precision} \times \text{Recall}}{0.8 \times \text{Precision} + 0.2 \times \text{Recall}}$$

Under some simplifying assumptions — specifically, that precision = recall — basic calculus shows that  $F_{0.8}$  is four times as sensitive to recall than to precision. Following the same reasoning, we want to place more emphasis on recall than on precision. We are also constrained by the linear programming framework, hence we set the objective function as

$$\frac{1}{Z} \left( \sum_X c_{\text{ref}}(X) + f \sum_Y c_{\text{cand}}(Y) \right) \quad 0 < f < 1$$



We set  $f = 0.25$  so that our objective function is also four times as sensitive to recall than to precision.<sup>3</sup> The value of this objective function is our TESLA-CELAB score. Similar to the other TESLA metrics, when there are  $N$  multiple references, we match the candidate translation against each of them and use the average of the  $N$  objective function values as the segment level score. System level score is the average of all the segment level scores.

$Z$  is a normalizing constant to scale the metric to the range  $[0, 1]$ , chosen so that when all the  $c(\cdot)$  variables have the value of one, our metric score attains the value of one.

## 4 Experiments

In this section, we test the effectiveness of TESLA-CELAB on some real-world English-Chinese translation tasks.

### 4.1 IWSLT 2008 English-Chinese CT

The test set of the IWSLT 2008 (Paul, 2008) English-Chinese ASR challenge task (CT) consists of 300 sentences of spoken language text. The average English source sentence is 5.8 words long and the average Chinese reference translation is 9.2 characters long. The domain is travel expressions.

The test set was translated by seven MT systems, and each translation has been manually judged for adequacy and fluency. Adequacy measures whether the translation conveys the correct meaning, even if the translation is not fully fluent, whereas fluency measures whether a translation is fluent, regardless of whether the meaning is correct. Due to high evaluation costs, adequacy and fluency assessments were limited to the translation outputs of four systems. In addition, the translation outputs of the MT systems are also manually ranked according to their translation quality.

Inter-judge agreement is measured by the Kappa coefficient, defined as:

$$\text{Kappa} = \frac{P(A) - P(E)}{1 - P(E)}$$

where  $P(A)$  is the percentage of agreement, and  $P(E)$  is the percentage of agreement by pure

<sup>3</sup>Our empirical experiments suggest that the correlations do plateau near this value. For simplicity, we choose not to tune  $f$  on the training data.

Judgment Set	2	3
1	0.4406	0.4355
2	-	0.4134

Table 1: Inter-judge Kappa for the NIST 2008 English-Chinese task

chance. The inter-judge Kappa is 0.41 for fluency, 0.40 for adequacy, and 0.57 for ranking. Kappa values between 0.4 and 0.6 are considered *moderate*, and the numbers are in line with other comparable experiments.

### 4.2 NIST 2008 English-Chinese MT Task

The NIST 2008 English-Chinese MT task consists of 127 documents with 1,830 segments, each with four reference translations and eleven automatic MT system translations. The data is available as LDC2010T01 from the Linguistic Data Consortium (LDC). The domain is newswire texts. The average English source sentence is 21.5 words long and the average Chinese reference translation is 43.2 characters long.

Since no manual evaluation is given for the data set, we recruited twelve bilingual judges to evaluate the first thirty documents for adequacy and fluency (355 segments for a total of  $355 \times 11 = 3,905$  translated segments). The final score of a sentence is the average of its adequacy and fluency scores. Each judge works on one quarter of the sentences so that each translation is judged by three judges. The judgments are concatenated to form three full sets of judgments.

We ignore judgments where two sentences are equal in quality, so that there are only two possible outcomes (X is better than Y; or Y is better than X), and  $P(E) = 1/2$ . The Kappa values are shown in Table 1. The values indicate moderate agreement, and are in line with other comparable experiments.

### 4.3 Baseline Metrics

#### 4.3.1 BLEU

Although word-level BLEU has often been found inferior to the new-generation metrics when the target language is English or other European languages, prior research has shown that character-level BLEU is highly competitive when the target language is Chinese (Li et al., 2011). Therefore, we

Metric	Type	Segment consistency	Pearson correlation	Spearman rank correlation
BLEU	character-level	0.7004	0.9130	0.9643
TESLA-M	word-level	0.6771	0.9167	0.8929
TESLA-CELAB–	character-level	0.7018	0.9229	0.9643
TESLA-CELAB	hybrid	0.7281*	0.9490**	0.9643

Table 2: Correlation with human judgment on the IWSLT 2008 English-Chinese challenge task. \* denotes better than the BLEU baseline at 5% significance level. \*\* denotes better than the BLEU baseline at 1% significance level.

Metric	Type	Segment consistency	Pearson correlation	Spearman rank correlation
BLEU	character-level	0.7091	0.8429	0.7818
TESLA-M	word-level	0.6969	0.8301	0.8091
TESLA-CELAB–	character-level	0.7158	0.8514	0.8227
TESLA-CELAB	hybrid	0.7162	0.8923**	0.8909**

Table 3: Correlation with human judgment on the NIST 2008 English-Chinese MT task. \*\* denotes better than the BLEU baseline at 1% significance level.

use character-level BLEU as our main baseline.

The correlations of character-level BLEU and the average human judgments are shown in the first row of Tables 2 and 3 for the IWSLT and the NIST data set, respectively. Segment-level consistency is defined as the number of correctly predicted pairwise rankings divided by the total number of pairwise rankings. Ties are excluded from the calculation. We also report the Pearson correlation and the Spearman rank correlation of the system-level scores. Note that in the IWSLT data set, the Spearman rank correlation is highly unstable due to the small number of participating systems.

#### 4.3.2 TESLA-M

In addition to character-level BLEU, we also present the correlations for the word-level metric TESLA. Compared to BLEU, TESLA allows more sophisticated weighting of n-grams and measures of word similarity including synonym relations. It has been shown to give better correlations than BLEU for many European languages including English (Callison-Burch et al., 2011). However, its use of POS tags and synonym dictionaries prevents its use at the character-level. We use TESLA as a representative of a competitive word-level metric.

We use the Stanford Chinese word segmenter (Tseng et al., 2005) and POS tagger (Toutanova et al., 2003) for preprocessing and Cilin for synonym

definition during matching. TESLA has several variants, and the simplest and often the most robust, TESLA-M, is used in this work. The various correlations are reported in the second row of Tables 2 and 3.

The scores show that word-level TESLA-M has no clear advantage over character-level BLEU, despite its use of linguistic features. We consider this conclusion to be in line with that of Li et al. (2011).

#### 4.4 TESLA-CELAB

In all our experiments here we use TESLA-CELAB with n-grams for  $n$  up to four, since the vast majority of Chinese words, and therefore synonyms, are at most four characters long.

The correlations between the TESLA-CELAB scores and human judgments are shown in the last row of Tables 2 and 3. We conducted significance testing using the resampling method of (Koehn, 2004). Entries that outperform the BLEU baseline at 5% significance level are marked with ‘\*’, and those that outperform at the 1% significance level are marked with ‘\*\*’. The results indicate that TESLA-CELAB significantly outperforms BLEU.

For comparison, we also run TESLA-CELAB without the use of the Cilin dictionary, reported in the third row of Tables 2 and 3 and denoted as TESLA-CELAB–. This disables TESLA-

CELAB’s ability to detect word-level synonyms and turns TESLA-CELAB into a linear programming based character-level metric. The performance of TESLA-CELAB– is comparable to the character-level BLEU baseline.

Note that

- TESLA-M can process word-level synonyms, but does not award character-level matches.
- TESLA-CELAB– and character-level BLEU award character-level matches, but do not consider word-level synonyms.
- TESLA-CELAB can process word-level synonyms and can award character-level matches.

Therefore, the difference between TESLA-M and TESLA-CELAB highlights the contribution of character-level matching, and the difference between TESLA-CELAB– and TESLA-CELAB highlights the contribution of word-level synonyms.

#### 4.5 Sample Sentences

Some sample sentences taken from the IWSLT test set are shown in Table 4 (some are simplified from the original). The Cilin dictionary correctly identified the following as synonyms:

周	=	星期	<i>week</i>
女儿	=	闺女	<i>daughter</i>
你	=	您	<i>you</i>
工作	=	上班	<i>work</i>

The dictionary fails to recognize the following synonyms:

一个	=	个	<i>a</i>
这儿	=	这里	<i>here</i>

However, partial awards are still given for the matching characters 这 and 个.

Based on these synonyms, TESLA-CELAB is able to award less trivial n-gram matches, such as 下周=下星期, 个女儿=个闺女, and 工作吗=上班吗, as these pairs can all be segmented into aligned synonyms. The covered n-gram matching rule is then able to award tricky n-grams such as 下星, 个女, 个闺, 作吗 and 班吗, which are covered by 下星期, 个女儿, 个闺女, 工作吗 and 上班吗 respectively.

Note also that the word segmentations shown in these examples are for clarity only. The TESLA-CELAB algorithm does not need pre-segmented

Reference:	下	周	。
	next	week	.
Candidate:	下	星期	。
	next	week	.

---

Reference:	我	有	一个	女儿	。
	I	have	a	daughter	.
Candidate:	我	有	个	闺女	。
	I	have	a	daughter	.

---

Reference:	你	在	这儿	工作	吗	？
	you	at	here	work	qn	?
Candidate:	您	在	这里	上班	吗	？
	you	at	here	work	qn	?

Table 4: Sample sentences from the IWSLT 2008 test set

Schirm	kaufen
<i>umbrella</i>	<i>buy</i>

---

Regenschirm	kaufen
<i>umbrella</i>	<i>buy</i>

---

Regen	schirm	kaufen
<i>rain</i>	<i>umbrella</i>	<i>buy</i>

Figure 5: Three forms of *buy umbrella* in German

sentences, and essentially finds multi-character synonyms opportunistically.

## 5 Discussion and Future Work

### 5.1 Other Languages with Ambiguous Word Boundaries

Although our experiments here are limited to Chinese, many other languages have similarly ambiguous word boundaries. For example, in German, the exact counterpart to our example exists, as depicted in Figure 5.

Regenschirm, literally *rain-umbrella*, is a synonym of Schirm. The first two forms in Figure 5 appear in natural text, and in standard BLEU, they would be penalized for the non-matching words *Schirm* and *Regenschirm*. Since compound nouns such as *Regenschirm* are very common in German and generate many out-of-vocabulary words, a common preprocessing step in German translation (and translation evaluation to a lesser extent) is to split compound words, and we end up with the last form *Regen schirm kaufen*. This process is analogous to

Chinese word segmentation.

We plan to conduct experiments on German and other Asian languages with the same linguistic phenomenon in future work.

## 5.2 Fractional Similarity Measures

In the current formulation of TESLA-CELAB, two n-grams  $X$  and  $Y$  are either synonyms which completely match each other, or are completely unrelated. In contrast, the linear-programming based TESLA metric allows fractional similarity measures between 0 (completely unrelated) and 1 (exact synonyms). We can then award partial scores for related words, such as those identified as such by WordNet or those with the same POS tags.

Supporting fractional similarity measures is non-trivial in the TESLA-CELAB framework. We plan to address this in future work.

## 5.3 Fractional Weights for N-grams

The TESLA-M metric allows each n-gram to have a weight, which is primarily used to discount function words. TESLA-CELAB can support fractional weights for n-grams as well by the following extension. We introduce a function  $m(X)$  that assigns a weight in  $[0, 1]$  for each n-gram  $X$ . Accordingly, our objective function is replaced by:

$$\frac{1}{Z} \left( \sum_X m(X)c_{\text{ref}}(X) + f \sum_Y m(Y)c_{\text{cand}}(Y) \right)$$

where  $Z$  is a normalizing constant so that the metric has a range of  $[0, 1]$ .

$$Z = \sum_X m(X) + f \sum_Y m(Y)$$

However, experiments with different weight functions  $m(\cdot)$  on the test data set failed to find a better weight function than the currently implied  $m(\cdot) = 1$ . This is probably due to the linguistic characteristics of Chinese, where human judges apparently give equal importance to function words and content words. In contrast, TESLA-M found discounting function words very effective for English and other European languages such as German. We plan to investigate this in the context of non-Chinese languages.

## 6 Conclusion

In this work, we devise a new MT evaluation metric in the family of TESLA (Translation Evaluation of Sentences with Linear-programming-based Analysis), called TESLA-CELAB (Character-level Evaluation for Languages with Ambiguous word Boundaries), to address the problem of fuzzy word boundaries in the Chinese language, although neither the phenomenon nor the method is unique to Chinese. Our metric combines the advantages of character-level and word-level metrics:

1. TESLA-CELAB is able to award scores for partial word-level matches.
2. TESLA-CELAB does not have a segmentation step, hence it will not introduce word segmentation errors.
3. TESLA-CELAB is able to take full advantage of the synonym dictionary, even when the synonyms differ in the number of characters.

We show empirically that TESLA-CELAB significantly outperforms the strong baseline of character-level BLEU in two well known English-Chinese MT evaluation data sets. The source code of TESLA-CELAB is available from <http://nlp.comp.nus.edu.sg/software/>.

## Acknowledgments

This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

## References

- Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 workshop on statistical machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*.

- Chris Callison-Burch, Philipp Koehn, Christof Monz, Kay Peterson, Mark Przybocki, and Omar F. Zaidan. 2010. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*.
- Chris Callison-Burch, Philipp Koehn, Christof Monz, and Omar Zaidan. 2011. Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.
- Yee Seng Chan and Hwee Tou Ng. 2008. MAXSIM: A maximum similarity metric for machine translation evaluation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Daniel Dahlmeier, Chang Liu, and Hwee Tou Ng. 2011. TESLA at WMT2011: Translation evaluation and tunable metric. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.
- Maoxi Li, Chengqing Zong, and Hwee Tou Ng. 2011. Automatic evaluation of Chinese translation output: word-level or character-level? In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Short Papers*.
- Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2010. TESLA: Translation evaluation of sentences with linear-programming-based analysis. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*.
- Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2011. Better evaluation metrics lead to better machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. 2005. A maximum entropy approach to Chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.
- Michael Paul. 2008. Overview of the iwslt 2008 evaluation campaign. In *Proceedings of the International Workshop on Spoken Language Translation*.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the Association for Machine Translation in the Americas*.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics*.
- Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for SIGHAN bakeoff 2005. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- Hongmei Zhao and Qun Liu. 2010. The CIPS-SIGHAN CLP 2010 Chinese word segmentation bakeoff. In *Proceedings of the Joint Conference on Chinese Language Processing*.

# PORT: a Precision-Order-Recall MT Evaluation Metric for Tuning

**Boxing Chen, Roland Kuhn and Samuel Larkin**

National Research Council Canada

283 Alexandre-Taché Boulevard, Gatineau (Québec), Canada J8X 3X7

{Boxing.Chen, Roland.Kuhn, Samuel.Larkin}@nrc.ca

## Abstract

Many machine translation (MT) evaluation metrics have been shown to correlate better with human judgment than BLEU. In principle, tuning on these metrics should yield better systems than tuning on BLEU. However, due to issues such as speed, requirements for linguistic resources, and optimization difficulty, they have not been widely adopted for tuning. This paper presents PORT<sup>1</sup>, a new MT evaluation metric which combines precision, recall and an ordering metric and which is primarily designed for tuning MT systems. PORT does not require external resources and is quick to compute. It has a better correlation with human judgment than BLEU. We compare PORT-tuned MT systems to BLEU-tuned baselines in five experimental conditions involving four language pairs. PORT tuning achieves consistently better performance than BLEU tuning, according to four automated metrics (including BLEU) and to human evaluation: in comparisons of outputs from 300 source sentences, human judges preferred the PORT-tuned output 45.3% of the time (*vs.* 32.7% BLEU tuning preferences and 22.0% ties).

## 1 Introduction

Automatic evaluation metrics for machine translation (MT) quality are a key part of building statistical MT (SMT) systems. They play two

roles: to allow rapid (though sometimes inaccurate) comparisons between different systems or between different versions of the same system, and to perform tuning of parameter values during system training. The latter has become important since the invention of minimum error rate training (MERT) (Och, 2003) and related tuning methods. These methods perform repeated decoding runs with different system parameter values, which are tuned to optimize the value of the evaluation metric over a development set with reference translations.

MT evaluation metrics fall into three groups:

- BLEU (Papineni *et al.*, 2002), NIST (Doddington, 2002), WER, PER, TER (Snover *et al.*, 2006), and LRscore (Birch and Osborne, 2011) do not use external linguistic information; they are fast to compute (except TER).
- METEOR (Banerjee and Lavie, 2005), METEOR-NEXT (Denkowski and Lavie 2010), TER-Plus (Snover *et al.*, 2009), MaxSim (Chan and Ng, 2008), TESLA (Liu *et al.*, 2010), AMBER (Chen and Kuhn, 2011) and MTeRater (Parton *et al.*, 2011) exploit some limited linguistic resources, such as synonym dictionaries, part-of-speech tagging, paraphrasing tables or word root lists.
- More sophisticated metrics such as RTE (Pado *et al.*, 2009), DCU-LFG (He *et al.*, 2010) and MEANT (Lo and Wu, 2011) use higher level syntactic or semantic analysis to score translations.

Among these metrics, BLEU is the most widely used for both evaluation and tuning. Many of the metrics correlate better with human judgments of translation quality than BLEU, as shown in recent WMT *Evaluation Task* reports (Callison-Burch *et*

---

<sup>1</sup> PORT: Precision-Order-Recall Tunable metric.

al., 2010; Callison-Burch *et al.*, 2011). However, BLEU remains the *de facto* standard tuning metric, for two reasons. First, there is no evidence that any other tuning metric yields better MT systems. Cer *et al.* (2010) showed that BLEU tuning is more robust than tuning with other metrics (METEOR, TER, *etc.*), as gauged by both automatic and human evaluation. Second, though a tuning metric should correlate strongly with human judgment, MERT (and similar algorithms) invoke the chosen metric so often that it must be computed quickly.

Liu *et al.* (2011) claimed that TESLA tuning performed better than BLEU tuning according to human judgment. However, in the WMT 2011 “tunable metrics” shared pilot task, this did not hold (Callison-Burch *et al.*, 2011). In (Birch and Osborne, 2011), humans preferred the output from LRscore-tuned systems 52.5% of the time, versus BLEU-tuned system outputs 43.9% of the time.

In this work, our goal is to devise a metric that, like BLEU, is computationally cheap and language-independent, but that yields better MT systems than BLEU when used for tuning. We tried out different combinations of statistics before settling on the final definition of our metric. The final version, PORT, combines precision, recall, strict brevity penalty (Chiang *et al.*, 2008) and strict redundancy penalty (Chen and Kuhn, 2011) in a quadratic mean expression. This expression is then further combined with a new measure of word ordering,  $v$ , designed to reflect long-distance as well as short-distance word reordering (BLEU only reflects short-distance reordering). In a later section, 3.3, we describe experiments that vary parts of the definition of PORT.

Results given below show that PORT correlates better with human judgments of translation quality than BLEU does, and sometimes outperforms METEOR in this respect, based on data from WMT (2008-2010). However, since PORT is designed for tuning, the most important results are those showing that PORT tuning yields systems with better translations than those produced by BLEU tuning – both as determined by automatic metrics (including BLEU), and according to human judgment, as applied to five data conditions involving four language pairs.

## 2 BLEU and PORT

First, define n-gram precision  $p(n)$  and recall  $r(n)$ :

$$p(n) = \frac{\# \text{n - grams}(T \cap R)}{\# \text{n - grams}(T)} \quad (1)$$

$$r(n) = \frac{\# \text{n - grams}(T \cap R)}{\# \text{n - grams}(R)} \quad (2)$$

where  $T$  = translation,  $R$  = reference. Both BLEU and PORT are defined on the document-level, *i.e.*  $T$  and  $R$  are whole texts. If there are multiple references, we use closest reference length for each translation hypothesis to compute the numbers of the reference n-grams.

### 2.1 BLEU

BLEU is composed of precision  $P_g(N)$  and brevity penalty  $BP$ :

$$BLEU = P_g(N) \times BP \quad (3)$$

where  $P_g(N)$  is the geometric average of n-gram precisions

$$P_g(N) = \left( \prod_{n=1}^N p(n) \right)^{\frac{1}{N}} \quad (4)$$

The BLEU brevity penalty punishes the score if the translation length  $len(T)$  is shorter than the reference length  $len(R)$ ; it is:

$$BP = \min(1.0, e^{1-len(R)/len(T)}) \quad (5)$$

### 2.2 PORT

PORT has five components: precision, recall, strict brevity penalty (Chiang *et al.*, 2008), strict redundancy penalty (Chen and Kuhn, 2011) and an ordering measure  $v$ . The design of PORT is based on exhaustive experiments on a development data set. We do not have room here to give a rationale for all the choices we made when we designed PORT. However, a later section (3.3) reconsiders some of these design decisions.

#### 2.2.1 Precision and Recall

The average precision and average recall used in PORT (unlike those used in BLEU) are the arithmetic average of n-gram precisions  $P_a(N)$  and recalls  $R_a(N)$ :

$$P_a(N) = \frac{1}{N} \sum_{n=1}^N p(n) \quad (6)$$

$$R_a(N) = \frac{1}{N} \sum_{n=1}^N r(n) \quad (7)$$

We use two penalties to avoid too long or too short MT outputs. The first, the strict brevity penalty (SBP), is proposed in (Chiang *et al.*, 2008). Let  $t_i$  be the translation of input sentence  $i$ , and let  $r_i$  be its reference. Set

$$SBP = \exp\left(1 - \frac{\sum_i |r_i|}{\sum_i \min\{|t_i|, |r_i|\}}\right) \quad (8)$$

The second is the strict redundancy penalty (SRP), proposed in (Chen and Kuhn, 2011):

$$SRP = \exp\left(1 - \frac{\sum_i \max\{|t_i|, |r_i|\}}{\sum_i |r_i|}\right) \quad (9)$$

To combine precision and recall, we tried four averaging methods: arithmetic (A), geometric (G), harmonic (H), and quadratic (Q) mean. If all of the values to be averaged are positive, the order is  $min \leq H \leq G \leq A \leq Q \leq max$ , with equality holding if and only if all the values being averaged are equal. We chose the quadratic mean to combine precision and recall, as follows:

$$Qmean(N) = \sqrt{\frac{(P_a(N) \times SBP)^2 + (R_a(N) \times SRP)^2}{2}} \quad (10)$$

### 2.2.2 Ordering Measure

Word ordering measures for MT compare two permutations of the original source-language word sequence: the permutation represented by the sequence of corresponding words in the MT output, and the permutation in the reference. Several ordering measures have been integrated into MT evaluation metrics recently. Birch and Osborne (2011) use either Hamming Distance or Kendall’s  $\tau$  Distance (Kendall, 1938) in their metric LRscore, thus obtaining two versions of LRscore. Similarly, Isozaki *et al.* (2011) adopt either Kendall’s  $\tau$  Distance or Spearman’s  $\rho$  (Spearman, 1904) distance in their metrics.

Our measure,  $v$ , is different from all of these. We use word alignment to compute the two permutations (LRscore also uses word alignment). The word alignment between the source input and reference is computed using GIZA++ (Och and Ney, 2003) beforehand with the default settings, then is refined with the heuristic *grow-diag-final-and*; the word alignment between the source input and the translation is generated by the decoder with the help of word alignment inside each phrase pair.

PORT uses permutations. These encode one-to-one relations but not one-to-many, many-to-one, many-to-many or null relations, all of which can occur in word alignments. We constrain the forbidden types of relation to become one-to-one, as in (Birch and Osborne, 2011). Thus, in a one-to-many alignment, the single source word is forced to align with the first target word; in a many-to-one alignment, monotone order is assumed for the target words; and source words originally aligned to null are aligned to the target word position just after the previous source word’s target position.

After the normalization above, suppose we have two permutations for the same source  $n$ -word input. *E.g.*, let  $P_1$  = reference,  $P_2$  = hypothesis:

$$P_1: p_1^1 \ p_1^2 \ p_1^3 \ p_1^4 \ \dots \ p_1^i \ \dots \ p_1^n$$

$$P_2: p_2^1 \ p_2^2 \ p_2^3 \ p_2^4 \ \dots \ p_2^i \ \dots \ p_2^n$$

Here, each  $p_i^j$  is an integer denoting position in the original source (*e.g.*,  $p_1^1 = 7$  means that the first word in  $P_1$  is the 7<sup>th</sup> source word).

The ordering metric  $v$  is computed from two distance measures. The first is absolute permutation distance:

$$DIST_1(P_1, P_2) = \sum_{i=1}^n |p_1^i - p_2^i| \quad (11)$$

$$\text{Let } v_1 = 1 - \frac{DIST_1(P_1, P_2)}{n(n+1)/2} \quad (12)$$

$v_1$  ranges from 0 to 1; a larger value means more similarity between the two permutations. This metric is similar to Spearman’s  $\rho$  (Spearman, 1904). However, we have found that  $\rho$  punishes long-distance reorderings too heavily. For instance,  $v_1$  is more tolerant than  $\rho$  of the movement of “recently” in this example:

Ref: *Recently, I visited Paris*

Hyp: *I visited Paris recently*

Inspired by HMM word alignment (Vogel *et al.*, 1996), our second distance measure is based on jump width. This punishes a sequence of words that moves a long distance with its internal order conserved, only once rather than on every word. In the following, only two groups of words have moved, so the jump width punishment is light:

Ref: *In the winter of 2010, I visited Paris*

Hyp: *I visited Paris in the winter of 2010*

So the second distance measure is



$$DIST_2(P_1, P_2) = \sum_{i=1}^n |(p_1^i - p_1^{i-1}) - (p_2^i - p_2^{i-1})| \quad (13)$$

where we set  $p_1^0 = 0$  and  $p_2^0 = 0$ . Let

$$v_2 = 1 - \frac{DIST_2(P_1, P_2)}{n^2 - 1} \quad (14)$$

As with  $v_1$ ,  $v_2$  is also from 0 to 1, and larger values indicate more similar permutations. The ordering measure  $v_s$  is the harmonic mean of  $v_1$  and  $v_2$ :

$$v_s = 2 / (1/v_1 + 1/v_2) \quad (15)$$

$v_s$  in (15) is computed at segment level. For multiple references, we compute  $v_s$  for each, and then choose the biggest one as the segment level ordering similarity. We compute document level ordering with a weighted arithmetic mean:

$$v = \frac{\sum_{s=1}^l v_s \times len_s(R)}{\sum_{s=1}^l len_s(R)} \quad (16)$$

where  $l$  is the number of segments of the document, and  $len(R)$  is the length of the reference.

### 2.2.3 Combined Metric

Finally,  $Qmean(N)$  (Eq. (10) and the word ordering measure  $v$  are combined in a harmonic mean:

$$PORT = \frac{2}{1/Qmean(N) + 1/v^\alpha} \quad (17)$$

Here  $\alpha$  is a free parameter that is tuned on held-out data. As it increases, the importance of the ordering measure  $v$  goes up. For our experiments, we tuned  $\alpha$  on Chinese-English data, setting it to 0.25 and keeping this value for the other language pairs. The use of  $v$  means that unlike BLEU, PORT requires word alignment information.

## 3 Experiments

### 3.1 PORT as an Evaluation Metric

We studied PORT as an evaluation metric on WMT data; test sets include WMT 2008, WMT 2009, and WMT 2010 all-to-English, plus 2009, 2010 English-to-all submissions. The languages “all” (“xx” in Table 1) include French, Spanish, German and Czech. Table 1 summarizes the test set statistics. In order to compute the  $v$  part of PORT, we require source-target word alignments for the references and MT outputs. These aren’t included in WMT data, so we compute them with GIZA++.

We used Spearman’s rank correlation coefficient  $\rho$  to measure correlation of the metric with system-level human judgments of translation. The human judgment score is based on the “Rank” only, *i.e.*, how often the translations of the system were rated as better than those from other systems (Callison-Burch *et al.*, 2008). Thus, BLEU, METEOR, and PORT were evaluated on how well their rankings correlated with the human ones. For the segment level, we follow (Callison-Burch *et al.*, 2010) in using Kendall’s rank correlation coefficient  $\tau$ .

As shown in Table 2, we compared PORT with smoothed BLEU (*mteval-v13a*), and METEOR v1.0. Both BLEU and PORT perform matching of  $n$ -grams up to  $n = 4$ .

Set	Year	Lang.	#system	#sent-pair
Test1	2008	xx-en	43	7,804
Test2	2009	xx-en	45	15,087
Test3	2009	en-xx	40	14,563
Test4	2010	xx-en	53	15,964
Test5	2010	en-xx	32	18,508

Table 1: Statistics of the WMT dev and test sets.

Metric	Into-En		Out-of-En	
	sys.	seg.	sys.	seg.
BLEU	0.792	0.215	0.777	0.240
METEOR	<b>0.834</b>	0.231	<b>0.835</b>	0.225
<b>PORT</b>	0.801	<b>0.236</b>	0.804	<b>0.242</b>

Table 2: Correlations with human judgment on WMT

PORT achieved the best segment level correlation with human judgment on both the “into English” and “out of English” tasks. At the system level, PORT is better than BLEU, but not as good as METEOR. This is because we designed PORT to carry out tuning; we did not optimize its performance as an evaluation metric, but rather, to optimize system tuning performance. There are some other possible reasons why PORT did not outperform METEOR v1.0 at system level. Most WMT submissions involve language pairs with similar word order, so the ordering factor  $v$  in PORT won’t play a big role. Also,  $v$  depends on source-target word alignments for reference and test sets. These alignments were performed by GIZA++ models trained on the test data only.

## 3.2 PORT as a Metric for Tuning

### 3.2.1 Experimental details

The first set of experiments to study PORT as a tuning metric involved Chinese-to-English (zh-en); there were two data conditions. The first is the *small data* condition where *FBIS*<sup>2</sup> is used to train the translation and reordering models. It contains 10.5M target word tokens. We trained two language models (LMs), which were combined loglinearly. The first is a 4-gram LM which is estimated on the target side of the texts used in the *large data* condition (below). The second is a 5-gram LM estimated on English *Gigaword*.

The *large data* condition uses training data from NIST<sup>3</sup> 2009 (Chinese-English track). All allowed bilingual corpora except *UN*, *Hong Kong Laws* and *Hong Kong Hansard* were used to train the translation model and reordering models. There are about 62.6M target word tokens. The same two LMs are used for *large data* as for *small data*, and the same development (“dev”) and test sets are also used. The dev set comprised mainly data from the NIST 2005 test set, and also some balanced-genre web-text from NIST. Evaluation was performed on NIST 2006 and 2008. Four references were provided for all dev and test sets.

The third data condition is a French-to-English (fr-en). The parallel training data is from Canadian Hansard data, containing 59.3M word tokens. We used two LMs in loglinear combination: a 4-gram LM trained on the target side of the parallel training data, and the English *Gigaword* 5-gram LM. The dev set has 1992 sentences; the two test sets have 2140 and 2164 sentences respectively. There is one reference for all dev and test sets.

The fourth and fifth conditions involve German-English Europarl data. This parallel corpus contains 48.5M German tokens and 50.8M English tokens. We translate both German-to-English (de-en) and English-to-German (en-de). The two conditions both use an LM trained on the target side of the parallel training data, and de-en also uses the English *Gigaword* 5-gram LM. News test 2008 set is used as dev set; News test 2009, 2010, 2011 are used as test sets. One reference is provided for all dev and test sets.

All experiments were carried out with  $\alpha$  in Eq. (17) set to 0.25, and involved only lowercase European-language text. They were performed with MOSES (Koehn *et al.*, 2007), whose decoder includes lexicalized reordering, translation models, language models, and word and phrase penalties. Tuning was done with n-best MERT, which is available in MOSES. In all tuning experiments, both BLEU and PORT performed lower case matching of  $n$ -grams up to  $n = 4$ . We also conducted experiments with tuning on a version of BLEU that incorporates SBP (Chiang *et al.*, 2008) as a baseline. The results of original IBM BLEU and BLEU with SBP were tied; to save space, we only report results for original IBM BLEU here.

### 3.2.2 Comparisons with automatic metrics

First, let us see if BLEU-tuning and PORT-tuning yield systems with different translations for the same input. The first row of Table 3 shows the percentage of identical sentence outputs for the two tuning types on test data. The second row shows the similarity of the two outputs at word-level (as measured by 1-TER): *e.g.*, for the two zh-en tasks, the two tuning types give systems whose outputs are about 25-30% different at the word level. By contrast, only about 10% of output words for fr-en differ for BLEU vs. PORT tuning.

	zh-en <i>small</i>	zh-en <i>large</i>	fr-en Hans	de-en WMT	en-de WMT
Same sent.	17.7%	13.5%	56.6%	23.7%	26.1%
1-TER	74.2	70.9	91.6	87.1	86.6

Table 3: Similarity of BLEU-tuned and PORT-tuned system outputs on test data.

Task	Tune	Evaluation metrics (%)			
		BLEU	MTR	1-TER	PORT
zh-en <i>small</i>	BLEU	26.8	55.2	38.0	49.7
	PORT	<b>27.2*</b>	<b>55.7</b>	38.0	<b>50.0</b>
zh-en <i>large</i>	BLEU	29.9	58.4	41.2	53.0
	PORT	<b>30.3*</b>	<b>59.0</b>	<b>42.0</b>	<b>53.2</b>
fr-en Hans	BLEU	38.8	<b>69.8</b>	54.2	57.1
	PORT	38.8	69.6	<b>54.6</b>	57.1
de-en WMT	BLEU	20.1	55.6	38.4	39.6
	PORT	<b>20.3</b>	<b>56.0</b>	38.4	<b>39.7</b>
en-de WMT	BLEU	13.6	43.3	30.1	31.7
	PORT	13.6	43.3	<b>30.7</b>	31.7

Table 4: Automatic evaluation scores on test data. \* indicates the results are significantly better than the baseline ( $p < 0.05$ ).

<sup>2</sup> LDC2003E14

<sup>3</sup> <http://www.nist.gov/speech/tests/mt>

Table 4 shows translation quality for BLEU- and PORT-tuned systems, as assessed by automatic metrics. We employed BLEU4, METEOR (v1.0), TER (v0.7.25), and the new metric PORT. In the table, TER scores are presented as 1-TER to ensure that for all metrics, higher scores mean higher quality. All scores are averages over the relevant test sets. There are twenty comparisons in the table. Among these, there is one case (French-English assessed with METEOR) where BLEU outperforms PORT, there are seven ties, and there are twelve cases where PORT is better. Table 3 shows that fr-en outputs are very similar for both tuning types, so the fr-en results are perhaps less informative than the others. Overall, PORT tuning has a striking advantage over BLEU tuning.

Both (Liu *et al.*, 2011) and (Cer *et al.*, 2011) showed that with MERT, if you want the best possible score for a system’s translations according to metric M, then you should tune with M. This doesn’t appear to be true when PORT and BLEU tuning are compared in Table 4. For the two Chinese-to-English tasks in the table, PORT tuning yields a better BLEU score than BLEU tuning, with significance at  $p < 0.05$ . We are currently investigating why PORT tuning gives higher BLEU scores than BLEU tuning for Chinese-English and German-English. In internal tests we have found no systematic difference in dev-set BLEUs, so we speculate that PORT’s emphasis on reordering yields models that generalize better for these two language pairs.

### 3.2.3 Human Evaluation

We conducted a human evaluation on outputs from BLEU- and PORT-tuned systems. The examples are randomly picked from all “to-English” conditions shown in Tables 3 & 4 (*i.e.*, all conditions except English-to-German).

We performed pairwise comparison of the translations produced by the system types as in (Callison-Burch *et al.*, 2010; Callison-Burch *et al.*, 2011). First, we eliminated examples where the reference had fewer than 10 words or more than 50 words, or where outputs of the BLEU-tuned and PORT-tuned systems were identical. The evaluators (colleagues not involved with this paper) objected to comparing two bad translations, so we then selected for human evaluation only translations that had high sentence-level (1-TER) scores. To be fair to both metrics, for each

condition, we took the union of examples whose BLEU-tuned output was in the top  $n\%$  of BLEU outputs and those whose PORT-tuned output was in the top  $n\%$  of PORT outputs (based on (1-TER)). The value of  $n$  varied by condition: we chose the top 20% of zh-en *small*, top 20% of en-de, top 50% of fr-en and top 40% of zh-en *large*. We then randomly picked 450 of these examples to form the manual evaluation set. This set was split into 15 subsets, each containing 30 sentences. The first subset was used as a common set; each of the other 14 subsets was put in a separate file, to which the common set is added. Each of the 14 evaluators received one of these files, containing 60 examples (30 unique examples and 30 examples shared with the other evaluators). Within each example, BLEU-tuned and PORT-tuned outputs were presented in random order.

After receiving the 14 annotated files, we computed Fleiss’s Kappa (Fleiss, 1971) on the common set to measure inter-annotator agreement,  $\kappa_{all}$ . Then, we excluded annotators one at a time to compute  $\kappa^i$  (Kappa score without  $i$ -th annotator, *i.e.*, from the other 13). Finally, we filtered out the files from the 4 annotators whose answers were most different from everybody else’s: *i.e.*, annotators with the biggest  $\kappa_{all} - \kappa^i$  values.

This left 10 files from 10 evaluators. We threw away the common set in each file, leaving 300 pairwise comparisons. Table 5 shows that the evaluators preferred the output from the PORT-tuned system 136 times, the output from the BLEU-tuned one 98 times, and had no preference the other 66 times. This indicates that there is a human preference for outputs from the PORT-tuned system over those from the BLEU-tuned system at the  $p < 0.01$  significance level (in cases where people prefer one of them).

PORT tuning seems to have a bigger advantage over BLEU tuning when the translation task is hard. Of the Table 5 language pairs, the one where PORT tuning helps most has the lowest BLEU in Table 4 (German-English); the one where it helps least in Table 5 has the highest BLEU in Table 4 (French-English). (Table 5 does not prove BLEU is superior to PORT for French-English tuning: statistically, the difference between 14 and 17 here is a tie). Maybe by picking examples for each condition that were the easiest for the system to translate (to make human evaluation easier), we

mildly biased the results in Table 5 against PORT tuning. Another possible factor is reordering. PORT differs from BLEU partly in modeling long-distance reordering more accurately; English and French have similar word order, but the other two language pairs don't. The results in section 3.3 (below) for Qmean, a version of PORT without word ordering factor  $\nu$ , suggest  $\nu$  may be defined suboptimally for French-English.

	PORT win	BLEU win	equal	total
zh-en	<b>19</b>	18	12	49
<i>small</i>	<b>38.8%</b>	36.7%	24.5%	
zh-en	<b>69</b>	46	36	151
<i>large</i>	<b>45.7%</b>	30.5%	23.8%	
fr-en	14	<b>17</b>	12	43
Hans	32.6%	<b>39.5%</b>	27.9%	
de-en	<b>34</b>	17	6	57
WMT	<b>59.7%</b>	29.8%	10.5%	
All	<b>136</b>	98	66	300
	<b>45.3%</b>	32.7%	22.0%	

Table 5: Human preference for outputs from PORT-tuned vs. BLEU-tuned system.

### 3.2.4 Computation time

A good tuning metric should run very fast; this is one of the advantages of BLEU. Table 6 shows the time required to score the 100-best hypotheses for the dev set for each data condition during MERT for BLEU and PORT in similar implementations. The average time of each iteration, including model loading, decoding, scoring and running MERT<sup>4</sup>, is in brackets. PORT takes roughly 1.5 – 2.5 as long to compute as BLEU, which is reasonable for a tuning metric.

	zh-en	zh-en	fr-en	de-en	en-de
	<i>small</i>	<i>large</i>	Hans	WMT	WMT
BLEU	3 (13)	3 (17)	2 (19)	2 (20)	2 (11)
PORT	5 (21)	5 (24)	4 (28)	5 (28)	4 (15)

Table 6: Time to score 100-best hypotheses (average time per iteration) in minutes.

### 3.2.5 Robustness to word alignment errors

PORT, unlike BLEU, depends on word alignments. How does quality of word alignment between source and reference affect PORT tuning? We created a dev set from Chinese Tree Bank

<sup>4</sup> Our experiments are run on a cluster. The average time for an iteration includes queuing, and the speed of each node is slightly different, so bracketed times are only for reference.

(CTB) hand-aligned data. It contains 588 sentences (13K target words), with one reference. We also ran GIZA++ to obtain its automatic word alignment, computed on CTB and FBIS. The AER of the GIZA++ word alignment on CTB is 0.32.

In Table 7, CTB is the dev set. The table shows tuning with BLEU, PORT with human word alignment (PORT + HWA), and PORT with GIZA++ word alignment (PORT + GWA); the condition is zh-en *small*. Despite the AER of 0.32 for automatic word alignment, PORT tuning works about as well with this alignment as for the gold standard CTB one. (The BLEU baseline in Table 7 differs from the Table 4 BLEU baseline because the dev sets differ).

Tune	BLEU	MTR	1-TER	PORT
BLEU	25.1	53.7	36.4	47.8
PORT + HWA	<b>25.3</b>	54.4	<b>37.0</b>	<b>48.2</b>
PORT + GWA	<b>25.3</b>	<b>54.6</b>	36.4	48.1

Table 7: PORT tuning - human & GIZA++ alignment

Task	Tune	BLEU	MTR	1-TER	PORT
zh-en <i>small</i>	BLEU	26.8	55.2	38.0	49.7
	PORT	<b>27.2</b>	<b>55.7</b>	38.0	<b>50.0</b>
	Qmean	26.8	55.3	<b>38.2</b>	49.8
zh-en <i>large</i>	BLEU	29.9	58.4	41.2	53.0
	PORT	<b>30.3</b>	<b>59.0</b>	<b>42.0</b>	<b>53.2</b>
	Qmean	30.2	58.5	41.8	53.1
fr-en Hans	BLEU	38.8	<b>69.8</b>	54.2	57.1
	PORT	38.8	69.6	<b>54.6</b>	57.1
	Qmean	38.8	<b>69.8</b>	<b>54.6</b>	57.1
de-en WMT	BLEU	20.1	55.6	<b>38.4</b>	39.6
	PORT	<b>20.3</b>	56.0	<b>38.4</b>	<b>39.7</b>
	Qmean	<b>20.3</b>	<b>56.3</b>	38.1	<b>39.7</b>
en-de WMT	BLEU	13.6	43.3	30.1	31.7
	PORT	13.6	43.3	<b>30.7</b>	31.7
	Qmean	13.6	<b>43.4</b>	30.3	31.7

Table 8: Impact of ordering measure  $\nu$  on PORT

## 3.3 Analysis

Now, we look at the details of PORT to see which of them are the most important. We do not have space here to describe all the details we studied, but we can describe some of them. *E.g.*, does the ordering measure  $\nu$  help tuning performance? To answer this, we introduce an intermediate metric. This is Qmean as in Eq. (10): PORT without the ordering measure. Table 8 compares tuning with BLEU, PORT, and Qmean. PORT outperforms Qmean on seven of the eight automatic scores shown for *small* and *large* Chinese-English.

However, for the European language pairs, PORT and Qmean seem to be tied. This may be because we optimized  $\alpha$  in Eq. (18) for Chinese-English, making the influence of word ordering measure  $v$  in PORT too strong for the European pairs, which have similar word order.

Measure  $v$  seems to help Chinese-English tuning. What would results be on that language pair if we were to replace  $v$  in PORT with another ordering measure? Table 9 gives a partial answer, with Spearman’s  $\rho$  and Kendall’s  $\tau$  replacing  $v$  with  $\rho$  or  $\tau$  in PORT for the zh-en *small* condition (CTB with human word alignment is the dev set). The original definition of PORT seems preferable.

Tune	BLEU	METEOR	1-TER
BLEU	25.1	53.7	36.4
PORT( $v$ )	<b>25.3</b>	<b>54.4</b>	<b>37.0</b>
PORT( $\rho$ )	25.1	54.2	36.3
PORT( $\tau$ )	25.1	54.0	36.0

Table 9: Comparison of the ordering measure: replacing  $v$  with  $\rho$  or  $\tau$  in PORT.

Task	Tune	ordering measures		
		$\rho$	$\tau$	$v$
NIST06	BLEU	0.979	0.926	0.915
	PORT	0.979	<b>0.928</b>	<b>0.917</b>
NIST08	BLEU	0.980	0.926	0.916
	PORT	<b>0.981</b>	<b>0.929</b>	<b>0.918</b>
CTB	BLEU	0.973	0.860	0.847
	PORT	<b>0.975</b>	<b>0.866</b>	<b>0.853</b>

Table 10: Ordering scores ( $\rho$ ,  $\tau$  and  $v$ ) for test sets NIST 2006, 2008 and CTB.

A related question is how much word ordering improvement we obtained from tuning with PORT. We evaluate Chinese-English word ordering with three measures: Spearman’s  $\rho$ , Kendall’s  $\tau$  distance as applied to two permutations (see section 2.2.2) and our own measure  $v$ . Table 10 shows the effects of BLEU and PORT tuning on these three measures, for three test sets in the zh-en *large* condition. Reference alignments for CTB were created by humans, while the NIST06 and NIST08 reference alignments were produced with GIZA++. A large value of  $\rho$ ,  $\tau$ , or  $v$  implies outputs have ordering similar to that in the reference. From the table, we see that the PORT-tuned system yielded better word order than the BLEU-tuned system in all nine combinations of test sets and ordering measures. The advantage of PORT tuning is

particularly noticeable on the most reliable test set: the hand-aligned CTB data.

What is the impact of the strict redundancy penalty on PORT? Note that in Table 8, even though Qmean has no ordering measure, it outperforms BLEU. Table 11 shows the BLEU brevity penalty (BP) and (number of matching 1- & 4- grams)/(number of total 1- & 4- grams) for the translations. The BLEU-tuned and Qmean-tuned systems generate similar numbers of matching n-grams, but Qmean-tuned systems produce fewer n-grams (thus, shorter translations). *E.g.*, for zh-en *small*, the BLEU-tuned system produced 44,677 1-grams (words), while the Qmean-trained system one produced 43,555 1-grams; both have about 32,000 1-grams matching the references. Thus, the Qmean translations have higher precision. We believe this is because of the strict redundancy penalty in Qmean. As usual, French-English is the outlier: the two outputs here are typically so similar that BLEU and Qmean tuning yield very similar n-gram statistics.

Task	Tune	1-gram	4-gram	BP
zh-en <i>small</i>	BLEU	32055/44677	4603/39716	0.967
	Qmean	31996/43555	4617/38595	0.962
zh-en <i>large</i>	BLEU	34583/45370	5954/40410	0.972
	Qmean	34369/44229	5987/39271	0.959
fr-en Hans	BLEU	28141/40525	8654/34224	0.983
	Qmean	28167/40798	8695/34495	0.990
de-en WMT	BLEU	42380/75428	5151/66425	1.000
	Qmean	42173/72403	5203/63401	0.968
en-de WMT	BLEU	30326/62367	2261/54812	1.000
	Qmean	30343/62092	2298/54537	0.997

Table 11: #matching-ngram/#total-ngram and BP score

## 4 Conclusions

In this paper, we have proposed a new tuning metric for SMT systems. PORT incorporates precision, recall, strict brevity penalty and strict redundancy penalty, plus a new word ordering measure  $v$ . As an evaluation metric, PORT performed better than BLEU at the system level and the segment level, and it was competitive with or slightly superior to METEOR at the segment level. Most important, our results show that PORT-tuned MT systems yield better translations than BLEU-tuned systems on several language pairs, according both to automatic metrics and human evaluations. In future work, we plan to tune the free parameter  $\alpha$  for each language pair.

## References

- S. Banerjee and A. Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of ACL Workshop on Intrinsic & Extrinsic Evaluation Measures for Machine Translation and/or Summarization.
- A. Birch and M. Osborne. 2011. Reordering Metrics for MT. In Proceedings of ACL.
- C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz and J. Schroeder. 2008. Further Meta-Evaluation of Machine Translation. In Proceedings of WMT.
- C. Callison-Burch, M. Osborne, and P. Koehn. 2006. Re-evaluating the role of BLEU in machine translation research. In Proceedings of EACL.
- C. Callison-Burch, P. Koehn, C. Monz, K. Peterson, M. Przybocki and O. Zaidan. 2010. Findings of the 2010 Joint Workshop on Statistical Machine Translation and Metrics for Machine Translation. In Proceedings of WMT.
- C. Callison-Burch, P. Koehn, C. Monz and O. Zaidan. 2011. Findings of the 2011 Workshop on Statistical Machine Translation. In Proceedings of WMT.
- D. Cer, D. Jurafsky and C. Manning. 2010. The Best Lexical Metric for Phrase-Based Statistical MT System Optimization. In Proceedings of NAACL.
- Y. S. Chan and H. T. Ng. 2008. MAXSIM: A maximum similarity metric for machine translation evaluation. In Proceedings of ACL.
- B. Chen and R. Kuhn. 2011. AMBER: A Modified BLEU, Enhanced Ranking Metric. In: Proceedings of WMT. Edinburgh, UK. July.
- D. Chiang, S. DeNeefe, Y. S. Chan, and H. T. Ng. 2008. Decomposability of translation metrics for improved evaluation and efficient algorithms. In Proceedings of EMNLP, pages 610–619.
- M. Denkowski and A. Lavie. 2010. Meteor-next and the meteor paraphrase tables: Improved evaluation support for five target languages. In Proceedings of the Joint Fifth Workshop on SMT and MetricsMATR, pages 314–317.
- G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In Proceedings of HLT.
- J. L. Fleiss. 1971. Measuring nominal scale agreement among many raters. In *Psychological Bulletin*, Vol. 76, No. 5 pp. 378–382.
- Y. He, J. Du, A. Way and J. van Genabith. 2010. The DCU dependency-based metric in WMT-MetricsMATR 2010. In Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR, pages 324–328.
- H. Isozaki, T. Hirao, K. Duh, K. Sudoh, H. Tsukada. 2010. Automatic Evaluation of Translation Quality for Distant Language Pairs. In Proceedings of EMNLP.
- M. Kendall. 1938. A New Measure of Rank Correlation. In *Biometrika*, 30 (1–2), pp. 81–89.
- 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 Proceedings of ACL, pp. 177–180, Prague, Czech Republic.
- A. Lavie and M. J. Denkowski. 2009. The METEOR metric for automatic evaluation of machine translation. *Machine Translation*, 23.
- C. Liu, D. Dahlmeier, and H. T. Ng. 2010. TESLA: Translation evaluation of sentences with linear-programming-based analysis. In Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR, pages 329–334.
- C. Liu, D. Dahlmeier, and H. T. Ng. 2011. Better evaluation metrics lead to better machine translation. In Proceedings of EMNLP.
- C. Lo and D. Wu. 2011. MEANT: An inexpensive, high-accuracy, semi-automatic metric for evaluating translation utility based on semantic roles. In Proceedings of ACL.
- F. J. Och. 2003. Minimum error rate training in statistical machine translation. In Proceedings of ACL-2003. Sapporo, Japan.
- F. J. Och and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. In *Computational Linguistics*, 29, pp. 19–51.
- S. Pado, M. Galley, D. Jurafsky, and C.D. Manning. 2009. Robust machine translation evaluation with entailment features. In Proceedings of ACL-IJCNLP.
- K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In Proceedings of ACL.
- K. Parton, J. Tetreault, N. Madnani and M. Chodorow. 2011. E-rating Machine Translation. In Proceedings of WMT.
- M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A Study of Translation Edit Rate

with Targeted Human Annotation. In Proceedings of Association for Machine Translation in the Americas.

- M. Snover, N. Madnani, B. Dorr, and R. Schwartz. 2009. Fluency, Adequacy, or HTER? Exploring Different Human Judgments with a Tunable MT Metric. In Proceedings of the Fourth Workshop on Statistical Machine Translation, Athens, Greece.
- C. Spearman. 1904. The proof and measurement of association between two things. In *American Journal of Psychology*, 15, pp. 72–101.
- S. Vogel, H. Ney, and C. Tillmann. 1996. HMM based word alignment in statistical translation. In Proceedings of COLING.

# Mixing Multiple Translation Models in Statistical Machine Translation

Majid Razmara<sup>1</sup>    George Foster<sup>2</sup>    Baskaran Sankaran<sup>1</sup>    Anoop Sarkar<sup>1</sup>

<sup>1</sup> Simon Fraser University, 8888 University Dr., Burnaby, BC, Canada

{razmara, baskaran, anoop}@sfu.ca

<sup>2</sup> National Research Council Canada, 283 Alexandre-Taché Blvd, Gatineau, QC, Canada

george.foster@nrc.gc.ca

## Abstract

Statistical machine translation is often faced with the problem of combining training data from many diverse sources into a single translation model which then has to translate sentences in a new domain. We propose a novel approach, ensemble decoding, which combines a number of translation systems dynamically at the decoding step. In this paper, we evaluate performance on a domain adaptation setting where we translate sentences from the medical domain. Our experimental results show that ensemble decoding outperforms various strong baselines including mixture models, the current state-of-the-art for domain adaptation in machine translation.

## 1 Introduction

Statistical machine translation (SMT) systems require large parallel corpora in order to be able to obtain a reasonable translation quality. In statistical learning theory, it is assumed that the training and test datasets are drawn from the same distribution, or in other words, they are from the same domain. However, bilingual corpora are only available in very limited domains and building bilingual resources in a new domain is usually very expensive. It is an interesting question whether a model that is trained on an existing large bilingual corpus in a specific domain can be adapted to another domain for which little parallel data is present. Domain adaptation techniques aim at finding ways to adjust an *out-of-domain* (OUT) model to represent a target domain (*in-domain* or IN).

Common techniques for model adaptation adapt two main components of contemporary state-of-the-art SMT systems: the language model and the translation model. However, language model adaptation is a more straight-forward problem compared to

translation model adaptation, because various measures such as perplexity of adapted language models can be easily computed on data in the target domain. As a result, language model adaptation has been well studied in various work (Clarkson and Robinson, 1997; Seymore and Rosenfeld, 1997; Bacchiani and Roark, 2003; Eck et al., 2004) both for speech recognition and for machine translation. It is also easier to obtain monolingual data in the target domain, compared to bilingual data which is required for translation model adaptation. In this paper, we focused on adapting only the translation model by fixing a language model for all the experiments. We expect domain adaptation for machine translation can be improved further by combining orthogonal techniques for translation model adaptation combined with language model adaptation.

In this paper, a new approach for adapting the translation model is proposed. We use a novel system combination approach called ensemble decoding in order to combine two or more translation models with the goal of constructing a system that outperforms all the component models. The strength of this system combination method is that the systems are combined in the decoder. This enables the decoder to pick the best hypotheses for each span of the input. The main applications of ensemble models are domain adaptation, domain mixing and system combination. We have modified *Kriya* (Sankaran et al., 2012), an in-house implementation of hierarchical phrase-based translation system (Chiang, 2005), to implement ensemble decoding using multiple translation models.

We compare the results of ensemble decoding with a number of baselines for domain adaptation. In addition to the basic approach of concatenation of in-domain and out-of-domain data, we also trained a log-linear mixture model (Foster and Kuhn, 2007)



as well as the linear mixture model of (Foster et al., 2010) for conditional phrase-pair probabilities over IN and OUT. Furthermore, within the framework of ensemble decoding, we study and evaluate various methods for combining translation tables.

## 2 Baselines

The natural baseline for model adaption is to concatenate the IN and OUT data into a single parallel corpus and train a model on it. In addition to this baseline, we have experimented with two more sophisticated baselines which are based on mixture techniques.

### 2.1 Log-Linear Mixture

Log-linear translation model (TM) mixtures are of the form:

$$p(\bar{e}|\bar{f}) \propto \exp\left(\sum_m^M \lambda_m \log p_m(\bar{e}|\bar{f})\right)$$

where  $m$  ranges over IN and OUT,  $p_m(\bar{e}|\bar{f})$  is an estimate from a component phrase table, and each  $\lambda_m$  is a weight in the top-level log-linear model, set so as to maximize dev-set BLEU using minimum error rate training (Och, 2003). We learn separate weights for relative-frequency and lexical estimates for both  $p_m(\bar{e}|\bar{f})$  and  $p_m(\bar{f}|\bar{e})$ . Thus, for 2 component models (from IN and OUT training corpora), there are  $4 * 2 = 8$  TM weights to tune. Whenever a phrase pair does not appear in a component phrase table, we set the corresponding  $p_m(\bar{e}|\bar{f})$  to a small epsilon value.

### 2.2 Linear Mixture

Linear TM mixtures are of the form:

$$p(\bar{e}|\bar{f}) = \sum_m^M \lambda_m p_m(\bar{e}|\bar{f})$$

Our technique for setting  $\lambda_m$  is similar to that outlined in Foster et al. (2010). We first extract a joint phrase-pair distribution  $\tilde{p}(\bar{e}, \bar{f})$  from the development set using standard techniques (HMM word alignment with grow-diag-and symmetrization (Koehn et al., 2003)). We then find the set of weights  $\hat{\lambda}$  that minimize the cross-entropy of the mixture  $p(\bar{e}|\bar{f})$  with respect to  $\tilde{p}(\bar{e}, \bar{f})$ :

$$\hat{\lambda} = \operatorname{argmax}_{\lambda} \sum_{\bar{e}, \bar{f}} \tilde{p}(\bar{e}, \bar{f}) \log \sum_m^M \lambda_m p_m(\bar{e}|\bar{f})$$

For efficiency and stability, we use the EM algorithm to find  $\hat{\lambda}$ , rather than L-BFGS as in (Foster et al., 2010). Whenever a phrase pair does not appear in a component phrase table, we set the corresponding  $p_m(\bar{e}|\bar{f})$  to 0; pairs in  $\tilde{p}(\bar{e}, \bar{f})$  that do not appear in at least one component table are discarded. We learn separate linear mixtures for relative-frequency and lexical estimates for both  $p(\bar{e}|\bar{f})$  and  $p(\bar{f}|\bar{e})$ . These four features then appear in the top-level model as usual – there is no runtime cost for the linear mixture.

## 3 Ensemble Decoding

Ensemble decoding is a way to combine the expertise of different models in one single model. The current implementation is able to combine hierarchical phrase-based systems (Chiang, 2005) as well as phrase-based translation systems (Koehn et al., 2003). However, the method can be easily extended to support combining a number of heterogeneous translation systems e.g. phrase-based, hierarchical phrase-based, and/or syntax-based systems. This section explains how such models can be combined during the decoding.

Given a number of translation models which are already trained and tuned, the ensemble decoder uses hypotheses constructed from all of the models in order to translate a sentence. We use the bottom-up CKY parsing algorithm for decoding. For each sentence, a CKY chart is constructed. The cells of the CKY chart are populated with appropriate rules from all the phrase tables of different components. As in the Hiero SMT system (Chiang, 2005), the cells which span up to a certain length (i.e. the maximum span length) are populated from the phrase tables and the rest of the chart uses *glue rules* as defined in (Chiang, 2005).

The rules suggested from the component models are combined in a single set. Some of the rules may be unique and others may be common with other component model rule sets, though with different scores. Therefore, we need to combine the scores of such common rules and assign a single score to

them. Depending on the mixture operation used for combining the scores, we would get different mixture scores. The choice of mixture operation will be discussed in Section 3.1.

Figure 1 illustrates how the CKY chart is filled with the rules. Each cell, covering a span, is populated with rules from all component models as well as from cells covering a sub-span of it.

In the typical log-linear model SMT, the posterior probability for each phrase pair  $(\bar{e}, \bar{f})$  is given by:

$$p(\bar{e} | \bar{f}) \propto \exp \left( \underbrace{\sum_i w_i \phi_i(\bar{e}, \bar{f})}_{\mathbf{w} \cdot \boldsymbol{\phi}} \right)$$

Ensemble decoding uses the same framework for each individual system. Therefore, the score of a phrase-pair  $(\bar{e}, \bar{f})$  in the ensemble model is:

$$p(\bar{e} | \bar{f}) \propto \exp \left( \underbrace{\mathbf{w}_1 \cdot \boldsymbol{\phi}_1}_{1^{st} \text{ model}} \oplus \underbrace{\mathbf{w}_2 \cdot \boldsymbol{\phi}_2}_{2^{nd} \text{ model}} \oplus \dots \right)$$

where  $\oplus$  denotes the mixture operation between two or more model scores.

### 3.1 Mixture Operations

Mixture operations receive two or more scores (probabilities) and return the mixture score (probability). In this section, we explore different options for mixture operation and discuss some of the characteristics of these mixture operations.

- **Weighted Sum (wsum):** in *wsum* the ensemble probability is proportional to the weighted sum of all individual model probabilities (i.e. linear mixture).

$$p(\bar{e} | \bar{f}) \propto \sum_m^M \lambda_m \exp(\mathbf{w}_m \cdot \boldsymbol{\phi}_m)$$

where  $m$  denotes the index of component models,  $M$  is the total number of them and  $\lambda_i$  is the weight for component  $i$ .

- **Weighted Max (wmax):** where the ensemble score is the weighted max of all model scores.

$$p(\bar{e} | \bar{f}) \propto \max_m (\lambda_m \exp(\mathbf{w}_m \cdot \boldsymbol{\phi}_m))$$

- **Model Switching (Switch):** in model switching, each cell in the CKY chart gets populated only by rules from one of the models and the other models' rules are discarded. This is based on the hypothesis that each component model is an expert on certain parts of sentence. In this method, we need to define a binary indicator function  $\delta(\bar{f}, m)$  for each span and component model to specify rules of which model to retain for each span.

$$\delta(\bar{f}, m) = \begin{cases} 1, & m = \operatorname{argmax}_{n \in M} \psi(\bar{f}, n) \\ 0, & \text{otherwise} \end{cases}$$

The criteria for choosing a model for each cell,  $\psi(\bar{f}, n)$ , could be based on:

- **Max:** for each cell, the model that has the highest weighted best-rule score wins:

$$\psi(\bar{f}, n) = \lambda_n \max_{\bar{e}} (\mathbf{w}_n \cdot \boldsymbol{\phi}_n(\bar{e}, \bar{f}))$$

- **Sum:** Instead of comparing only the scores of the best rules, the model with the highest weighted sum of the probabilities of the rules wins. This sum has to take into account the translation table limit (*ttl*), on the number of rules suggested by each model for each cell:

$$\psi(\bar{f}, n) = \lambda_n \sum_{\bar{e}} \exp(\mathbf{w}_n \cdot \boldsymbol{\phi}_n(\bar{e}, \bar{f}))$$

The probability of each phrase-pair  $(\bar{e}, \bar{f})$  is computed as:

$$p(\bar{e} | \bar{f}) = \sum_m^M \delta(\bar{f}, m) p_m(\bar{e} | \bar{f})$$

- **Product (prod):** in Product models or Product of Experts (Hinton, 1999), the probability of the ensemble model or a rule is computed as the product of the probabilities of all components (or equally the sum of log-probabilities, i.e. log-linear mixture). Product models can also make use of weights to control the contribution of each component. These models are



explicit gradient information for the objective function. Component weights for each mixture operation are optimized on the dev-set using CONDOR.

## 4 Experiments & Results

### 4.1 Experimental Setup

We carried out translation experiments using the European Medicines Agency (EMA) corpus (Tiedemann, 2009) as IN, and the Europarl (EP) corpus<sup>1</sup> as OUT, for French to English translation. The dev and test sets were randomly chosen from the EMA corpus.<sup>2</sup> The details of datasets used are summarized in Table 1.

Dataset	Sents	Words	
		French	English
<b>EMA</b>	11770	168K	144K
<b>Europarl</b>	1.3M	40M	37M
<b>Dev</b>	1533	29K	25K
<b>Test</b>	1522	29K	25K

Table 1: Training, dev and test sets for EMA.

For the mixture baselines, we used a standard one-pass phrase-based system (Koehn et al., 2003), Portage (Sadat et al., 2005), with the following 7 features: relative-frequency and lexical translation model (TM) probabilities in both directions; word-displacement distortion model; language model (LM) and word count. The corpus was word-aligned using both HMM and IBM2 models, and the phrase table was the union of phrases extracted from these separate alignments, with a length limit of 7. It was filtered to retain the top 20 translations for each source phrase using the TM part of the current log-linear model.

For ensemble decoding, we modified an in-house implementation of hierarchical phrase-based system, *Kriya* (Sankaran et al., 2012) which uses the same features mentioned in (Chiang, 2005): forward and backward relative-frequency and lexical TM probabilities; LM; word, phrase and glue-rules penalty. GIZA++(Och and Ney, 2000) has been used for word alignment with phrase length limit of 7.

In both systems, feature weights were optimized using MERT (Och, 2003) and with a 5-gram lan-

<sup>1</sup>[www.statmt.org/europarl](http://www.statmt.org/europarl)

<sup>2</sup>Please contact the authors to access the data-sets.

guage model and Kneser-Ney smoothing was used in all the experiments. We used SRILM (Stolcke, 2002) as the language model toolkit. Fixing the language model allows us to compare various translation model combination techniques.

### 4.2 Results

Table 2 shows the results of the baselines. The first group are the baseline results on the phrase-based system discussed in Section 2 and the second group are those of our hierarchical MT system. Since the Hiero baselines results were substantially better than those of the phrase-based model, we also implemented the best-performing baseline, linear mixture, in our Hiero-style MT system and in fact it achieves the highest BLEU score among all the baselines as shown in Table 2. This baseline is run three times the score is averaged over the BLEU scores with standard deviation of 0.34.

Baseline	PBS	Hiero
<b>IN</b>	31.84	33.69
<b>OUT</b>	24.08	25.32
<b>IN + OUT</b>	31.75	33.76
<b>LOGLIN</b>	32.21	–
<b>LINMIX</b>	33.81	<b>35.57</b>

Table 2: The results of various baselines implemented in a phrase-based (PBS) and a Hiero SMT on EMA.

Table 3 shows the results of ensemble decoding with different mixture operations and model weight settings. Each mixture operation has been evaluated on the test-set by setting the component weights uniformly (denoted by *uniform*) and by tuning the weights using CONDOR (denoted by *tuned*) on a held-out set. The tuned scores (3rd column in Table 3) are averages of three runs with different initial points as in Clark et al. (2011). We also reported the BLEU scores when we applied the span-wise normalization heuristic. All of these mixture operations were able to significantly improve over the concatenation baseline. In particular, *Switching:Max* could gain up to 2.2 BLEU points over the concatenation baseline and 0.39 BLEU points over the best performing baseline (i.e. linear mixture model implemented in Hiero) which is statistically significant based on Clark et al. (2011) ( $p = 0.02$ ).

*Prod* when using with uniform weights gets the

Mixture Operation	Uniform	Tuned	Norm.
WMAX	35.39	35.47 (s=0.03)	35.47
WSUM	35.35	35.53 (s=0.04)	35.45
SWITCHING:MAX	35.93	<b>35.96</b> (s=0.01)	32.62
SWITCHING:SUM	34.90	34.72 (s=0.23)	34.90
PROD	33.93	35.24 (s=0.05)	35.02

Table 3: The results of ensemble decoding on EMEA for Fr2En when using uniform weights, tuned weights and normalization heuristic. The tuned BLEU scores are averaged over three runs with multiple initial points, as in (Clark et al., 2011), with the standard deviations in brackets .

lowest score among the mixture operations, however after tuning, it learns to bias the weights towards one of the models and hence improves by 1.31 BLEU points. Although *Switching:Sum* outperforms the concatenation baseline, it is substantially worse than other mixture operations. One explanation that *Switching:Max* is the best performing operation and *Switching:Sum* is the worst one, despite their similarities, is that *Switching:Max* prefers more peaked distributions while *Switching:Sum* favours a model that has fewer hypotheses for each span.

An interesting observation based on the results in Table 3 is that uniform weights are doing reasonably well given that the component weights are not optimized and therefore model scores may not be in the same scope (refer to discussion in §3.2). We suspect this is because a single LM is shared between both models. This shared component controls the variance of the weights in the two models when combined with the standard L-1 normalization of each model’s weights and hence prohibits models to have too varied scores for the same input. Though, it may not be the case when multiple LMs are used which are not shared.

Two sample sentences from the EMEA test-set along with their translations by the IN, OUT and Ensemble models are shown in Figure 2. The boxes show how the Ensemble model is able to use n-grams from the IN and OUT models to construct a better translation than both of them. In the first example, there are two OOVs one for each of the IN and OUT models. Our approach is able to resolve the OOV issues by taking advantage of the other model’s presence. Similarly, the second example shows how ensemble decoding improves lexical choices as well as word re-orderings.

## 5 Related Work

### 5.1 Domain Adaptation

Early approaches to domain adaptation involved information retrieval techniques where sentence pairs related to the target domain were retrieved from the training corpus using IR methods (Eck et al., 2004; Hildebrand et al., 2005). Foster et al. (2010), however, uses a different approach to select related sentences from OUT. They use language model perplexities from IN to select relevant sentences from OUT. These sentences are used to enrich the IN training set.

Other domain adaptation methods involve techniques that distinguish between general and domain-specific examples (Daumé and Marcu, 2006). Jiang and Zhai (2007) introduce a general instance weighting framework for model adaptation. This approach tries to penalize misleading training instances from OUT and assign more weight to IN-like instances than OUT instances. Foster et al. (2010) propose a similar method for machine translation that uses features to capture degrees of generality. Particularly, they include the output from an SVM classifier that uses the intersection between IN and OUT as positive examples. Unlike previous work on instance weighting in machine translation, they use phrase-level instances instead of sentences.

A large body of work uses interpolation techniques to create a single TM/LM from interpolating a number of LMs/TMs. Two famous examples of such methods are linear mixtures and log-linear mixtures (Koehn and Schroeder, 2007; Civera and Juan, 2007; Foster and Kuhn, 2007) which were used as baselines and discussed in Section 2. Other methods include using self-training techniques to exploit monolingual in-domain data (Ueffing et al., 2007;

SOURCE	aménorrhée , menstruations irrégulières
REF	amenorrhoea , irregular menstruation
IN	amenorrhoea , menstruations irrégulières
OUT	aménorrhée , irregular menstruation
ENSEMBLE	amenorrhoea , irregular menstruation

SOURCE	le traitement par naglazyme doit être supervisé par un médecin ayant l' expérience de la prise en charge des patients atteints de mps vi ou d' une autre maladie métabolique héréditaire .
REF	naglazyme treatment should be supervised by a physician experienced in the management of patients with mps vi or other inherited metabolic diseases .
IN	naglazyme treatment should be supervisé by a doctor the with in the management of patients with mps vi or other hereditary metabolic disease .
OUT	naglazyme 's treatment must be supervised by a doctor with the experience of the care of patients with mps vi. or another disease hereditary metabolic .
ENSEMBLE	naglazyme treatment should be supervised by a physician experienced in the management of patients with mps vi or other hereditary metabolic disease .

Figure 2: Examples illustrating how this method is able to use expertise of both out-of-domain and in-domain systems.

Bertoldi and Federico, 2009). In this approach, a system is trained on the parallel OUT and IN data and it is used to translate the monolingual IN data set. Iteratively, most confident sentence pairs are selected and added to the training corpus on which a new system is trained.

## 5.2 System Combination

Tackling the model adaptation problem using system combination approaches has been experimented in various work (Koehn and Schroeder, 2007; Hildebrand and Vogel, 2009). Among these approaches are sentence-based, phrase-based and word-based output combination methods. In a similar approach, Koehn and Schroeder (2007) use a feature of the factored translation model framework in Moses SMT system (Koehn and Schroeder, 2007) to use multiple alternative decoding paths. Two decoding paths, one for each translation table (IN and OUT), were used during decoding. The weights are set with minimum error rate training (Och, 2003).

Our work is closely related to Koehn and Schroeder (2007) but uses a different approach to deal with multiple translation tables. The Moses SMT system implements (Koehn and Schroeder,

2007) and can treat multiple translation tables in two different ways: *intersection* and *union*. In *intersection*, for each span only the hypotheses would be used that are present in all phrase tables. For each set of hypothesis with the same source and target phrases, a new hypothesis is created whose feature-set is the union of feature sets of all corresponding hypotheses. *Union*, on the other hand, uses hypotheses from all the phrase tables. The feature set of these hypotheses are expanded to include one feature set for each table. However, for the corresponding feature values of those phrase-tables that did not have a particular phrase-pair, a default log probability value of 0 is assumed (Bertoldi and Federico, 2009) which is counter-intuitive as it boosts the score of hypotheses with phrase-pairs that do not belong to all of the translation tables.

Our approach is different from Koehn and Schroeder (2007) in a number of ways. Firstly, unlike the multi-table support of Moses which only supports phrase-based translation table combination, our approach supports ensembles of both hierarchical and phrase-based systems. With little modification, it can also support ensemble of syntax-based systems with the other two state-of-the-art SMT sys-

tems. Secondly, our combining method uses the *union* option, but instead of preserving the features of all phrase-tables, it only combines their scores using various mixture operations. This enables us to experiment with a number of different operations as opposed to sticking to only one combination method. Finally, by avoiding increasing the number of features we can add as many translation models as we need without serious performance drop. In addition, MERT would not be an appropriate optimizer when the number of features increases a certain amount (Chiang et al., 2008).

Our approach differs from the model combination approach of DeNero et al. (2010), a generalization of consensus or minimum Bayes risk decoding where the search space consists of those of multiple systems, in that model combination uses forest of derivations of all component models to do the combination. In other words, it requires all component models to fully decode each sentence, compute n-gram expectations from each component model and calculate posterior probabilities over translation derivations. While, in our approach we only use partial hypotheses from component models and the derivation forest is constructed by the ensemble model. A major difference is that in the model combination approach the component search spaces are conjoined and they are not intermingled as opposed to our approach where these search spaces are intermixed on spans. This enables us to generate new sentences that cannot be generated by component models. Furthermore, various combination methods can be explored in our approach. Finally, main techniques used in this work are orthogonal to our approach such as Minimum Bayes Risk decoding, using n-gram features and tuning using MERT.

Finally, our work is most similar to that of Liu et al. (2009) where max-derivation and max-translation decoding have been used. Max-derivation finds a derivation with highest score and max-translation finds the highest scoring translation by summing the score of all derivations with the same yield. The combination can be done in two levels: translation-level and derivation-level. Their derivation-level max-translation decoding is similar to our ensemble decoding with *wsum* as the mixture operation. We did not restrict ourself to this particular mixture operation and experimented with a

number of different mixing techniques and as Table 3 shows we could improve over *wsum* in our experimental setup. Liu et al. (2009) used a modified version of MERT to tune max-translation decoding weights, while we use a two-step approach using MERT for tuning each component model separately and then using CONDOR to tune component weights on top of them.

## 6 Conclusion & Future Work

In this paper, we presented a new approach for domain adaptation using ensemble decoding. In this approach a number of MT systems are combined at decoding time in order to form an ensemble model. The model combination can be done using various mixture operations. We showed that this approach can gain up to 2.2 BLEU points over its concatenation baseline and 0.39 BLEU points over a powerful mixture model.

Future work includes extending this approach to use multiple translation models with multiple language models in ensemble decoding. Different mixture operations can be investigated and the behaviour of each operation can be studied in more details. We will also add capability of supporting syntax-based ensemble decoding and experiment how a phrase-based system can benefit from syntax information present in a syntax-aware MT system. Furthermore, ensemble decoding can be applied on domain mixing settings in which development sets and test sets include sentences from different domains and genres, and this is a very suitable setting for an ensemble model which can adapt to new domains at test time. In addition, we can extend our approach by applying some of the techniques used in other system combination approaches such as consensus decoding, using *n*-gram features, tuning using forest-based MERT, among other possible extensions.

## Acknowledgments

This research was partially supported by an NSERC, Canada (RGPIN: 264905) grant and a Google Faculty Award to the last author. We would like to thank Philipp Koehn and the anonymous reviewers for their valuable comments. We also thank the developers of GIZA++ and Condor which we used for our experiments.

## References

- M. Bacchiani and B. Roark. 2003. Unsupervised language model adaptation. In *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, volume 1, pages I-224 – I-227 vol.1, april.
- Nicola Bertoldi and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the Fourth Workshop on Statistical Machine Translation, StatMT '09*, pages 182–189, Stroudsburg, PA, USA. ACL.
- David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*. ACL.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270, Morristown, NJ, USA. ACL.
- Jorge Civera and Alfons Juan. 2007. Domain adaptation in statistical machine translation with mixture modelling. In *Proceedings of the Second Workshop on Statistical Machine Translation, StatMT '07*, pages 177–180, Stroudsburg, PA, USA. ACL.
- Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: controlling for optimizer instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2, HLT '11*, pages 176–181. ACL.
- P. Clarkson and A. Robinson. 1997. Language model adaptation using mixtures and an exponentially decaying cache. In *Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '97)-Volume 2 - Volume 2, ICASSP '97*, pages 799–, Washington, DC, USA. IEEE Computer Society.
- Hal Daumé, III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *J. Artif. Int. Res.*, 26:101–126, May.
- John DeNero, Shankar Kumar, Ciprian Chelba, and Franz Och. 2010. Model combination for machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10*, pages 975–983, Stroudsburg, PA, USA. ACL.
- Matthias Eck, Stephan Vogel, and Alex Waibel. 2004. Language model adaptation for statistical machine translation based on information retrieval. In *In Proceedings of LREC*.
- George Foster and Roland Kuhn. 2007. Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation, StatMT '07*, pages 128–135, Stroudsburg, PA, USA. ACL.
- George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pages 451–459, Stroudsburg, PA, USA. ACL.
- Almut Silja Hildebrand and Stephan Vogel. 2009. CMU system combination for WMT'09. In *Proceedings of the Fourth Workshop on Statistical Machine Translation, StatMT '09*, pages 47–50, Stroudsburg, PA, USA. ACL.
- Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the 10th EAMT 2005*, Budapest, Hungary, May.
- Geoffrey E. Hinton. 1999. Products of experts. In *Artificial Neural Networks, 1999. ICANN 99. Ninth International Conference on (Conf. Publ. No. 470)*, volume 1, pages 1–6.
- Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in nlp. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic, June. ACL.
- Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation, StatMT '07*, pages 224–227, Stroudsburg, PA, USA. ACL.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference of the NAACL*, pages 127–133, Edmonton, May. NAACL.
- Yang Liu, Haitao Mi, Yang Feng, and Qun Liu. 2009. Joint decoding with multiple translation models. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2, ACL '09*, pages 576–584, Stroudsburg, PA, USA. ACL.
- F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the ACL*, pages 440–447, Hongkong, China, October.
- Franz Josef Och. 2003. Minimum error rate training for statistical machine translation. In *Proceedings of the 41th Annual Meeting of the ACL*, Sapporo, July. ACL.



- Slav Petrov. 2010. Products of random latent variable grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 19–27, Stroudsburg, PA, USA. ACL.
- Fatiha Sadat, Howard Johnson, Akakpo Agbago, George Foster, Joel Martin, and Aaron Tikuisis. 2005. Portage: A phrase-based machine translation system. In *In Proceedings of the ACL Workshop on Building and Using Parallel Texts*, Ann Arbor. ACL.
- Baskaran Sankaran, Majid Razmara, and Anoop Sarkar. 2012. Kriya – an end-to-end hierarchical phrase-based mt system. *The Prague Bulletin of Mathematical Linguistics*, 97(97), April.
- Kristie Seymore and Ronald Rosenfeld. 1997. Using story topics for language model adaptation. In George Kokkinakis, Nikos Fakotakis, and Evangelos Dermatas, editors, *EUROSPEECH*. ISCA.
- Andrew Smith, Trevor Cohn, and Miles Osborne. 2005. Logarithmic opinion pools for conditional random fields. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 18–25, Stroudsburg, PA, USA. ACL.
- Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings International Conference on Spoken Language Processing*, pages 257–286.
- Jorg Tiedemann. 2009. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia.
- Nicola Ueffing, Gholamreza Haffari, and Anoop Sarkar. 2007. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 25–32, Prague, Czech Republic, June. ACL.
- Frank Vanden Berghen and Hugues Bersini. 2005. CONDOR, a new parallel, constrained extension of powell’s UOBYQA algorithm: Experimental results and comparison with the DFO algorithm. *Journal of Computational and Applied Mathematics*, 181:157–175, September.

# Hierarchical Chunk-to-String Translation\*

Yang Feng<sup>†</sup> Dongdong Zhang<sup>‡</sup> Mu Li<sup>‡</sup> Ming Zhou<sup>‡</sup> Qun Liu<sup>\*</sup>

<sup>†</sup> Department of Computer Science  
University of Sheffield  
Sheffield, UK  
y.feng@shef.ac.uk

<sup>‡</sup> Microsoft Research Asia  
dozhang@microsoft.com  
mul@microsoft.com  
mingzhou@microsoft.com

\*Key Laboratory of Intelligent Information Processing  
Institute of Computing Technology  
Chinese Academy of Sciences  
liuqun@ict.ac.cn

## Abstract

We present a hierarchical chunk-to-string translation model, which can be seen as a compromise between the hierarchical phrase-based model and the tree-to-string model, to combine the merits of the two models. With the help of shallow parsing, our model learns rules consisting of words and *chunks* and meanwhile introduce syntax cohesion. Under the weighed synchronous context-free grammar defined by these rules, our model searches for the best translation derivation and yields target translation simultaneously. Our experiments show that our model significantly outperforms the hierarchical phrase-based model and the tree-to-string model on English-Chinese Translation tasks.

## 1 Introduction

The hierarchical phrase-based model (Chiang, 2007) makes an advance of statistical machine translation by employing hierarchical phrases, which not only uses phrases to learn local translations but also uses hierarchical phrases to capture reorderings of words and subphrases which can cover a large scope. Besides, this model is formal syntax-based and does not need to specify the syntactic constituents of subphrases, so it can directly learn synchronous context-free grammars (SCFG) from a parallel text without relying on any linguistic annotations or assumptions, which makes it used conveniently and widely.

---

<sup>\*</sup>This work was done when the first author visited Microsoft Research Asia as an intern.

However, it is often desirable to consider syntactic constituents of subphrases, e.g. the hierarchical phrase

$$X \rightarrow \langle X_1 \text{ for } X_2, X_2 \text{ de } X_1 \rangle$$

can be applied to both of the following strings in Figure 1

“A request for a purchase of shares”  
“filed for bankruptcy”,

and get the following translation, respectively

“goumai gufen de shenqing”  
“pochan de shenqing”.

In the former, “A request” is a NP and this rule acts correctly while in the latter “filed” is a VP and this rule gives a wrong reordering. If we specify the first  $X$  on the right-hand side to NP, this kind of errors can be avoided.

The tree-to-string model (Liu et al., 2006; Huang et al., 2006) introduces linguistic syntax via source parse to direct word reordering, especially long-distance reordering. Furthermore, this model is formalised as Tree Substitution Grammars, so it observes syntactic cohesion. Syntactic cohesion means that the translation of a string covered by a subtree in a source parse tends to be continuous. Fox (2002) shows that translation between English and French satisfies cohesion in the majority cases. Many previous works show promising results with an assumption that syntactic cohesion explains almost all translation movement for some language pairs (Wu, 1997; Yamada and Knight, 2001; Eisner, 2003; Graehl and Knight, 2004; Quirk et al., 2005; Cherry, 2008; Feng et al., 2010).

But unfortunately, the tree-to-string model requires each node must be strictly matched during rule matching, which makes it strongly dependent on the relationship of tree nodes and their roles in the whole sentence. This will lead to data sparseness and being vulnerable to parse errors.

In this paper, we present a hierarchical chunk-to-string translation model to combine the merits of the two models. Instead of parse trees, our model introduces linguistic information in the form of *chunks*, so it does not need to care the internal structures and the roles in the main sentence of chunks. Based on shallow parsing results, it learns rules consisting of either words (terminals) or chunks (nonterminals), where adjacent chunks are packed into one nonterminal. It searches for the best derivation through the SCFG-motivated space defined by these rules and get target translation simultaneously. In some sense, our model can be seen as a compromise between the hierarchical phrase-based model and the tree-to-string model, specifically

- Compared with the hierarchical phrase-based model, it integrates linguistic syntax and satisfies syntactic cohesion.
- Compared with the tree-to-string model, it only needs to perform shallow parsing which introduces less parsing errors. Besides, our model allows a nonterminal in a rule to cover several chunks, which can alleviate data sparseness and the influence of parsing errors.
- we refine our hierarchical chunk-to-string model into two models: a loose model (Section 2.1) which is more similar to the hierarchical phrase-based model and a tight model (Section 2.2) which is more similar to the tree-to-string model.

The experiments show that on the 2008 NIST English-Chinese MT translation test set, both the loose model and the tight model outperform the hierarchical phrase-based model and the tree-to-string model, where the loose model has a better performance. While in terms of speed, the tight model runs faster and its speed ranking is between the tree-to-string model and the hierarchical phrase-based model.

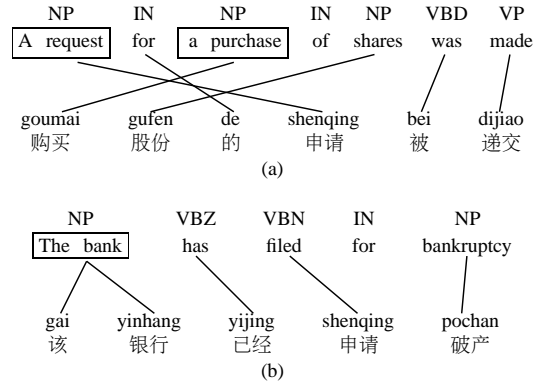
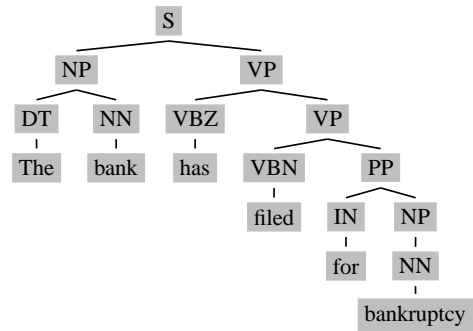


Figure 1: A running example of two sentences. For each sentence, the first row gives the chunk sequence.



(a) A parse tree

B-NP I-NP B-VBZ B-VBN B-IN B-NP  
The bank has filed for bankruptcy

(b) A chunk sequence got from the parse tree

Figure 2: An example of shallow parsing.

## 2 Modeling

**Shallow parsing** (also **chunking**) is an analysis of a sentence which identifies the constituents (noun groups, verbs, verb groups, etc), but neither specifies their internal structures, nor their roles in the main sentence. In Figure 1, we give the chunk sequence in the first row for each sentence. We treat shallow parsing as a sequence label task, and a sentence  $f$  can have many possible different chunk label sequences. Therefore, in theory, the conditional probability of a target translation  $e$  conditioned on the source sentence  $f$  is given by taking the chunk label sequences as a latent variable  $c$ :

$$p(e|f) = \sum_c p(c|f)p(e|f, c) \quad (1)$$

In practice, we only take the best chunk label sequence  $\hat{c}$  got by

$$\hat{c} = \operatorname{argmax}_c p(c|\mathbf{f}) \quad (2)$$

Then we can ignore the conditional probability  $p(\hat{c}|\mathbf{f})$  as it holds the same value for each translation, and get:

$$\begin{aligned} p(e|\mathbf{f}) &= p(\hat{c}|\mathbf{f})p(e|\mathbf{f}, \hat{c}) \\ &= p(e|\mathbf{f}, \hat{c}) \end{aligned} \quad (3)$$

We formalize our model as a weighted SCFG. In a SCFG, each rule (usually called production in SCFGs) has an aligned pair of right-hand sides — the source side and the target side, just as follows:

$$X \rightarrow \langle \alpha, \beta, \sim \rangle$$

where  $X$  is a nonterminal,  $\alpha$  and  $\beta$  are both strings of terminals and nonterminals, and  $\sim$  denotes one-to-one links between nonterminal occurrences in  $\alpha$  and nonterminal occurrences in  $\beta$ . A SCFG produces a derivation by starting with a pair of start symbols and recursively rewrites every two coindexed nonterminals with the corresponding components of a matched rule. A derivation yields a pair of strings on the right-hand side which are translation of each other.

In a weighted SCFG, each rule has a weight and the total weight of a derivation is the production of the weights of the rules used by the derivation. A translation may be produced by many different derivations and we only use the best derivation to evaluate its probability. With  $d$  denoting a derivation and  $r$  denoting a rule, we have

$$\begin{aligned} p(e|\mathbf{f}) &= \max_d p(d, e|\mathbf{f}, \hat{c}) \\ &= \max_d \prod_{r \in d} p(r, e|\mathbf{f}, \hat{c}) \end{aligned} \quad (4)$$

Following Och and Ney (2002), we frame our model as a log-linear model:

$$p(e|\mathbf{f}) = \frac{\exp \sum_k \lambda_k H_k(d, e, \hat{c}, \mathbf{f})}{\exp \sum_{d', e', k} \lambda_k H_k(d', e', \hat{c}, \mathbf{f})} \quad (5)$$

$$\text{where } H_k(d, e, \hat{c}, \mathbf{f}) = \sum_r h_k(\mathbf{f}, \hat{c}, r)$$

So the best translation is given by

$$\hat{e} = \operatorname{argmax}_e \sum_k \lambda_k H_k(d, e, \hat{c}, \mathbf{f}) \quad (6)$$

We employ the same set of features for the log-linear model as the hierarchical phrase-based model does (Chiang, 2005).

We further refine our hierarchical chunk-to-string model into two models: a loose model which is more similar to the hierarchical phrase-based model and a tight model which is more similar to the tree-to-string model. The two models differ in the form of rules and the way of estimating rule probabilities. While for decoding, we employ the same decoding algorithm for the two models: given a test sentence, the decoders first perform shallow parsing to get the best chunk sequence, then apply a CYK parsing algorithm with beam search.

## 2.1 A Loose Model

In our model, we employ rules containing nonterminals to handle long-distance reordering where boundary words play an important role. So for the subphrases which cover more than one chunk, we just maintain boundary chunks: we bundle adjacent chunks into one nonterminal and denote it as the first chunk tag immediately followed by “-” and next followed by the last chunk tag. Then, for the string pair  $\langle \text{filed for bankruptcy, shenqing pochan} \rangle$ , we can get the rule

$$r_1 : X \rightarrow \langle \text{VBN}_{\boxed{1}} \text{ for NP}_{\boxed{2}}, \text{VBN}_{\boxed{1}} \text{ NP}_{\boxed{2}} \rangle$$

while for the string pair  $\langle \text{A request for a purchase of shares, goumai gufen de shenqing} \rangle$ , we can get

$$r_2 : X \rightarrow \langle \text{NP}_{\boxed{1}} \text{ for NP-NP}_{\boxed{2}}, \text{NP-NP}_{\boxed{2}} \text{ de NP}_{\boxed{1}} \rangle.$$

The rule matching “A request for a purchase of shares was” will be

$$r_3 : X \rightarrow \langle \text{NP-NP}_{\boxed{1}} \text{ VBD}_{\boxed{2}}, \text{NP-NP}_{\boxed{1}} \text{ VBD}_{\boxed{2}} \rangle.$$

We can see that in contrast to the method of representing each chunk separately, this representation form can alleviate data sparseness and the influence of parsing errors.

$\langle S_{[1]}, S_{[1]} \rangle \Rightarrow \langle S_{[1]} X_{[1]}, S_{[1]} X_{[1]} \rangle$   
 $\Rightarrow \langle X_{[1]} X_{[1]}, X_{[1]} X_{[1]} \rangle$   
 $\Rightarrow \langle NP-NP_{[1]} VBD_{[1]} X_{[1]}, NP-NP_{[1]} VBD_{[1]} X_{[1]} \rangle$   
 $\Rightarrow \langle NP_{[1]} \text{ for } NP-NP_{[1]} VBD_{[1]} X_{[1]}, NP-NP_{[1]} \text{ de } NP_{[1]} VBD_{[1]} X_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for } NP-NP_{[1]} VBD_{[1]} X_{[1]}, NP-NP_{[1]} \text{ de shenqing } VBD_{[1]} X_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for a purchase of shares } VBD_{[1]} X_{[1]}, \text{ goumai gufen de shenqing } VBD_{[1]} X_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for a purchase of shares was } X_{[1]}, \text{ goumai gufen de shenqing bei } X_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for a purchase of shares was made, goumai gufen de shenqing bei dijiao} \rangle$

(a) The loose model

$\langle NP-VP_{[1]}, NP-VP_{[1]} \rangle \Rightarrow \langle NP-VBD_{[1]} VP_{[1]}, NP-VBD_{[1]} VP_{[1]} \rangle$   
 $\Rightarrow \langle NP-NP_{[1]} VBD_{[1]} VP_{[1]}, NP-NP_{[1]} VBD_{[1]} VP_{[1]} \rangle$   
 $\Rightarrow \langle NP_{[1]} \text{ for } NP-NP_{[1]} VBD_{[1]} VP_{[1]}, NP-NP_{[1]} \text{ de } NP_{[1]} VBD_{[1]} VP_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for } NP-NP_{[1]} VBD_{[1]} VP_{[1]}, NP-NP_{[1]} \text{ de shenqing } VBD_{[1]} VP_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for a purchase of shares } VBD_{[1]} VP_{[1]}, \text{ goumai gufen de shenqing } VBD_{[1]} VP_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for a purchase of shares was } VP_{[1]}, \text{ goumai gufen de shenqing bei } VP_{[1]} \rangle$   
 $\Rightarrow \langle \text{A request for a purchase of shares was made, goumai gufen de shenqing bei dijiao} \rangle$

(b) The tight model

Figure 3: The derivations of the sentence in Figure 1(a).

In these rules, the left-hand nonterminal symbol  $X$  can not match any nonterminal symbol on the right-hand side. So we need a set of rules such as

$$\begin{aligned}
 NP &\rightarrow \langle X_{[1]}, X_{[1]} \rangle \\
 NP-NP &\rightarrow \langle X_{[1]}, X_{[1]} \rangle
 \end{aligned}$$

and so on, and set the probabilities of these rules to 1. To simplify the derivation, we discard this kind of rules and assume that  $X$  can match any nonterminal on the right-hand side.

Only with  $r_2$  and  $r_3$ , we cannot produce any derivation of the whole sentence in Figure 1 (a). In this case we need two special *glue rules*:

$$\begin{aligned}
 r_4 : \quad S &\rightarrow \langle S_{[1]} X_{[2]}, S_{[1]} X_{[2]} \rangle \\
 r_5 : \quad S &\rightarrow \langle X_{[1]}, X_{[1]} \rangle
 \end{aligned}$$

Together with the following four lexical rules,

$$\begin{aligned}
 r_6 : \quad X &\rightarrow \langle \text{a request, shenqing} \rangle \\
 r_7 : \quad X &\rightarrow \langle \text{a purchase of shares, goumai gufen} \rangle \\
 r_8 : \quad X &\rightarrow \langle \text{was, bei} \rangle \\
 r_9 : \quad X &\rightarrow \langle \text{made, dijiao} \rangle
 \end{aligned}$$

Figure 3(a) shows the derivation of the sentence in Figure 1(a).

## 2.2 A Tight Model

In the tight model, the right-hand side of each rule remains the same as the loose model, but the left-hand side nonterminal is not  $X$  but the corresponding chunk labels. If a rule covers more than one chunk, we just use the first and the last chunk labels to denote the left-hand side nonterminal. The rule set used in the tight model for the example in Figure 1(a) corresponding to that in the loose model becomes:

$$\begin{aligned}
 r_2 : \quad NP-NP &\rightarrow \langle NP_{[1]} \text{ for } NP-NP_{[2]}, NP-NP_{[2]} \text{ de } NP_{[1]} \rangle \\
 r_3 : \quad NP-VBD &\rightarrow \langle NP-NP_{[1]} VBD_{[2]}, NP-NP_{[1]} VBD_{[2]} \rangle.
 \end{aligned}$$

$$\begin{aligned}
 r_6 : \quad NP &\rightarrow \langle \text{a request, shenqing} \rangle \\
 r_7 : \quad NP-NP &\rightarrow \langle \text{a purchase of shares, goumai gufen} \rangle \\
 r_8 : \quad VBD &\rightarrow \langle \text{was, bei} \rangle \\
 r_9 : \quad VP &\rightarrow \langle \text{made, dijiao} \rangle
 \end{aligned}$$

During decoding, we first collect rules for each span. For a span which does not have any matching rule, if we do not construct default rules for it, there will be no derivation for the whole sentence, then we need to construct default rules for this kind of span by enumerating all possible binary segmentation of the chunks in this span. For the example in Figure 1(a), there is no rule matching the whole sentence,

so we need to construct default rules for it, which should be

$$\text{NP-VP} \rightarrow \langle \text{NP-VBD}_{\square} \text{VP}_{\square}, \text{NP-VBD}_{\square} \text{VP}_{\square} \rangle.$$

$$\text{NP-VP} \rightarrow \langle \text{NP-NP}_{\square} \text{VBD-VP}_{\square}, \text{NP-NP}_{\square} \text{VBD-VP}_{\square} \rangle.$$

and so on.

Figure 3(b) shows the derivation of the sentence in Figure 1(a).

### 3 Shallow Parsing

In a parse tree, a chunk is defined by a leaf node or an inner node whose children are all leaf nodes (See Figure 2 (a)). In our model, we identify chunks by traversing a parse tree in a breadth-first order. Once a node is recognized as a chunk, we skip its children. In this way, we can get a sole chunk sequence given a parse tree. Then we label each word with a label indicating whether the word starts a chunk (B-) or continues a chunk (I-). Figure 2(a) gives an example. In this method, we get the training data for shallow parsing from Penn Tree Bank.

We take shallow Parsing (chunking) as a sequence label task and employ Conditional Random Field (CRF)<sup>1</sup> to train a chunker. CRF is a good choice for label tasks as it can avoid label bias and use more statistical correlated features. We employ the features described in Sha and Pereira (2003) for CRF. We do not introduce CRF-based chunkier in this paper and more details can be got from Hammersley and Clifford (1971), Lafferty et al. (2001), Taskar et al. (2002), Sha and Pereira (2003).

### 4 Rule Extraction

In what follows, we introduce how to get the rule set. We learn rules from a corpus that first is bi-directionally word-aligned by the GIZA++ toolkit (Och and Ney, 2000) and then is refined using a “final-and” strategy. We generate the rule set in two steps: first, we extract two sets of phrases, *basic phrases* and *chunk-based phrases*. Basic phrases are defined using the same heuristic as previous systems (Koehn et al., 2003; Och and Ney, 2004; Chiang, 2005). A chunk-based phrase is such a basic phrase that covers one or more chunks on the source side.

<sup>1</sup>We use the open source toolkit CRF++ got in <http://code.google.com/p/crfpp/>.

We identify chunk-based phrases  $\langle c_{j_1}^{j_2}, f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$  as follows:

1. A chunk-based phrase is a basic phrase;
2.  $c_{j_1}$  begins with “B-”;
3.  $f_{j_2}$  is the end word on the source side or  $c_{j_2+1}$  does not begins with “I-”.

Given a sentence pair  $\langle f, e, \sim \rangle$ , we extract rules for the loose model as follows

1. If  $\langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$  is a basic phrase, then we can have a rule

$$X \rightarrow \langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$$

2. Assume  $X \rightarrow \langle \alpha, \beta \rangle$  is a rule with  $\alpha = \alpha_1 f_{j_1}^{j_2} \alpha_2$  and  $\beta = \beta_1 e_{i_1}^{i_2} \beta_2$ , and  $\langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$  is a chunk-based phrase with a chunk sequence  $Y_u \cdots Y_v$ , then we have the following rule

$$X \rightarrow \langle \alpha_1 Y_u - Y_{v[\square]} \alpha_2, \beta_1 Y_u - Y_{v[\square]} \beta_2 \rangle.$$

We evaluate the distribution of these rules in the same way as Chiang (2007).

We extract rules for the tight model as follows

1. If  $\langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$  is a chunk-based phrase with a chunk sequence  $Y_s \cdots Y_t$ , then we can have a rule

$$Y_s - Y_t \rightarrow \langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$$

2. Assume  $Y_s - Y_t \rightarrow \langle \alpha, \beta \rangle$  is a rule with  $\alpha = \alpha_1 f_{j_1}^{j_2} \alpha_2$  and  $\beta = \beta_1 e_{i_1}^{i_2} \beta_2$ , and  $\langle f_{j_1}^{j_2}, e_{i_1}^{i_2} \rangle$  is a chunk-based phrase with a chunk sequence  $Y_u \cdots Y_v$ , then we have the following rule

$$Y_s - Y_t \rightarrow \langle \alpha_1 Y_u - Y_{v[\square]} \alpha_2, \beta_1 Y_u - Y_{v[\square]} \beta_2 \rangle.$$

We evaluate the distribution of rules in the same way as Liu et al. (2006).

For the loose model, the nonterminals must be cohesive, while the whole rule can be noncohesive: if both ends of a rule are nonterminals, the whole rule is cohesive, otherwise, it may be noncohesive. In contrast, for the tight model, both the whole rule and the nonterminal are cohesive.

Even with the cohesion constraints, our model still generates a large number of rules, but not all

of the rules are useful for translation. So we follow the method described in Chiang (2007) to filter the rule set except that we allow two nonterminals to be adjacent.

## 5 Related Works

Watanabe et al. (2003) presented a chunk-to-string translation model where the decoder generates a translation by first translating the words in each chunk, then reordering the translation of chunks. Our model distinguishes from their model mainly in reordering model. Their model reorders chunks resorting to a distortion model while our model reorders chunks according to SCFG rules which retain the relative positions of chunks.

Nguyen et al. (2008) presented a tree-to-string phrase-based method which is based on SCFGs. This method generates SCFGs through syntactic transformation including a word-to-phrase tree transformation model and a phrase reordering model while our model learns SCFG-based rules from word-aligned bilingual corpus directly

There are also some works aiming to introduce linguistic knowledge into the hierarchical phrase-based model. Marton and Resnik (2008) took the source parse tree into account and added soft constraints to hierarchical phrase-based model. Cherry (2008) used dependency tree to add syntactic cohesion. These methods work with the original SCFG defined by hierarchical phrase-based model and use linguistic knowledge to assist translation. Instead, our model works under the new defined SCFG with chunks.

Besides, some other researchers make efforts on the tree-to-string model by employing exponentially alternative parses to alleviate the drawback of 1-best parse. Mi et al. (2008) presented forest-based translation where the decoder translates a packed forest of exponentially many parses instead of i-best parse. Liu and Liu (2010) proposed to parse and to translate jointly by taking tree-based translation as parsing. Given a source sentence, this decoder produces a parse tree on the source side and a translation on the target side simultaneously. Both the models perform in the unit of tree nodes rather than chunks.

## 6 Experiments

### 6.1 Data Preparation

**Data for shallow parsing** We got training data and test data for shallow parsing from the standard Penn Tree Bank (PTB) English parsing task by splitting the sections 02-21 on the Wall Street Journal Portion (Marcus et al., 1993) into two sets: the last 1000 sentences as the test set and the rest as the training set. We filtered the features whose frequency was lower than 3 and substituted `` and '' with ~ to keep consistent with translation data. We used  $L2$  algorithm to train CRF.

**Data for Translation** We used the NIST training set for Chinese-English translation tasks excluding the Hong Kong Law and Hong Kong Hansard<sup>2</sup> as the training data, which contains 470K sentence pairs. For the training data set, we first performed word alignment in both directions using GIZA++ toolkit (Och and Ney, 2000) then refined the alignments using “final-and”. We trained a 5-gram language model with modified Kneser-Ney smoothing on the Xinhua portion of LDC Chinese Gigaword corpus. For the tree-to-string model, we parsed English sentences using Stanford parser and extracted rules using the GHKM algorithm (Galley et al., 2004).

We used our in-house English-Chinese data set as the development set and used the 2008 NIST English-Chinese MT test set (1859 sentences) as the test set. Our evaluation metric was BLEU-4 (Papineni et al., 2002) based on characters (as the target language is Chinese), which performed case-insensitive matching of n-grams up to  $n = 4$  and used the shortest reference for the brevity penalty. We used the standard minimum error-rate training (Och, 2003) to tune the feature weights to maximize the BLEU score on the development set.

### 6.2 Shallow Parsing

The standard evaluation metrics for shallow parsing are precision P, recall R, and their harmonic mean F1 score, given by:

$$P = \frac{\text{number of exactly recognized chunks}}{\text{number of output chunks}}$$
$$R = \frac{\text{number of exactly recognized chunks}}{\text{number of reference chunks}}$$

<sup>2</sup>The source side and target side are reversed.

Word number	Chunk number	Accuracy %
23861	12258	94.48

Chunk type	P %	R %	F <sub>1</sub> %	Found
All	91.14	91.35	91.25	12286
One	90.32	90.99	90.65	5236
NP	93.97	94.47	94.22	5523
ADVP	82.53	84.30	83.40	475
VP	93.66	92.04	92.84	284
ADJP	65.68	69.20	67.39	236
WHNP	96.30	95.79	96.04	189
QP	83.06	80.00	81.50	183

Table 1: Shallow parsing result. The column *Found* gives the number of chunks recognized by CRF, the row *All* represents all types of chunks, and the row *One* represents the chunks that consist of one word.

$$F_1 = \frac{2 \cdot P \cdot R}{P + R}$$

Besides, we need another metric, accuracy *A*, to evaluate the accurate rate of individual labeling decisions of every word as

$$A = \frac{\text{number of exactly labeled words}}{\text{number of words}}$$

For example, given a reference sequence B-NP I-NP I-NP B-VP I-VP B-VP, CRF outputs a sequence O-NP I-NP I-NP B-VP I-VP I-NP, then  $P = 33.33\%$ ,  $A = 66.67\%$ .

Table 1 summarizes the results of shallow parsing. For ‘‘ and ‘’ were substituted with ‘’, the performance was slightly influenced.

The F1 score of all chunks is 91.25% and the F1 score of *One* and NP, which in number account for about 90% of chunks, is 90.65% and 94.22% respectively. F score of NP chunking approaches 94.38% given in Sha and Pereira (2003).

### 6.3 Performance Comparison

We compared our loose decoder and tight decoder with our in-house hierarchical phrase-based decoder (Chiang, 2007) and the tree-to-string decoder (Liu et al., 2006). We set the same configuration for all the decoders as follows: stack size = 30, nbest size = 30. For the hierarchical chunk-based and phrase-based decoders, we set max rule length to 5. For the tree-to-string decoder, we set the configuration of rule

System	Dev	NIST08	Speed
phrase	0.2843	0.3921	1.163
tree	0.2786	0.3817	1.107
tight	0.2914	0.3987	1.208
loose	0.2936	0.4023	1.429

Table 2: Performance comparison. *Phrase* represents the hierarchical phrase-based decoder, *tree* represents the tree-to-string decoder, *tight* represents our tight decoder and *loose* represents our loose decoder. The speed is reported by seconds per sentence. The speed for the tree-to-string decoder includes the parsing time (0.23s) and the speed for the tight and loose models includes the shallow parsing time, too.

extraction as: the height up to 3 and the number of leaf nodes up to 5.

We give the results in Table 2. From the results, we can see that both the loose and tight decoders outperform the baseline decoders and the improvement is significant using the *sign-test* of Collins et al. (2005) ( $p < 0.01$ ). Specifically, the loose model has a better performance while the tight model has a faster speed.

Compared with the hierarchical phrase-based model, the loose model only imposes syntactic cohesion to nonterminals while the tight model imposes syntax cohesion to both rules and nonterminals which reduces search space, so it decodes faster. We can conclude that linguistic syntax can indeed improve the translation performance; syntactic cohesion for nonterminals can explain linguistic phenomena well; noncohesive rules are useful, too. The extra time consumption against hierarchical phrase-based system comes from shallow parsing.

By investigating the translation result, we find that our decoder does well in rule selection. For example, in the hierarchical phrase-based model, this kind of rules, such as

$$X \rightarrow \langle X \text{ of } X, * \rangle, X \rightarrow \langle X \text{ for } X, * \rangle$$

and so on, where  $*$  stands for the target component, are used with a loose restriction as long as the terminals are matched, while our models employ more stringent constraints on these rules by specifying the syntactic constituent of ‘‘X’’. With chunk labels, our models can make different treatment for different situations.



System	Dev	NIST08	Speed
cohesive	0.2936	0.4023	1.429
noncohesive	0.2937	0.3964	1.734

Table 3: Influence of cohesion. The row *cohesive* represents the loose system where nonterminals satisfy cohesion, and the row *noncohesive* represents the modified version of the loose system where nonterminals can be noncohesive.

Compared with the tree-to-string model, the result indicates that the change of the source-side linguistic syntax from parses to chunks can improve translation performance. The reasons should be our model can reduce parse errors and it is enough to use chunks as the basic unit for machine translation. Although our decoders and tree-to-string decoder all run in linear-time with beam search, tree-to-string model runs faster for it searches through a smaller SCFG-motivated space.

#### 6.4 Influence of Cohesion

We verify the influence of syntax cohesion via the loose model. The cohesive model imposes syntax cohesion on nonterminals to ensure the chunk is reordered as a whole. In this experiment, we introduce a noncohesive model by allowing a nonterminal to match part of a chunk. For example, in the noncohesive model, it is legal for a rule with the source side

“NP for NP-NP”

to match

“request for a purchase of shares”

in Figure 1 (a), where “request” is part of NP. As well, the rule with the source side

“NP for a NP-NP”

can match

“request for a purchase of shares”.

In this way, we can ensure all the rules used in the cohesive system can be used in the noncohesive system. Besides cohesive rules, the noncohesive system can use noncohesive rules, too.

We give the results in Table 3. From the results, we can see that cohesion helps to reduce search space, so the cohesive system decodes faster. The noncohesive system decoder slower, as it employs

System	Number	Dev	NIST08	Speed
loose	two	0.2936	0.4023	1.429
loose	three	0.2978	0.4037	2.056
tight	two	0.2914	0.3987	1.208
tight	three	0.2954	0.4026	1.780

Table 4: The influence of the number of nonterminals. The column *number* lists the number of nonterminals used at most in a rule.

more rules, but this does not bring any improvement of translation performance. As other researches said in their papers, syntax cohesion can explain linguistic phenomena well.

#### 6.5 Influence of the number of nonterminals

We also tried to allow a rule to hold three nonterminals at most. We give the result in Table 4. The result shows that using three nonterminals does not bring a significant improvement of translation performance but quite more time consumption. So we only retain two nonterminals at most in a rule.

### 7 Conclusion

In this paper, we present a hierarchical chunk-to-string model for statistical machine translation which can be seen as a compromise of the hierarchical phrase-based model and the tree-to-string model. With the help of shallow parsing, our model learns rules consisting of either words or chunks and compresses adjacent chunks in a rule to a nonterminal, then it searches for the best derivation under the SCFG defined by these rules. Our model can combine the merits of both the models: employing linguistic syntax to direct decoding, being syntax cohesive and robust to parsing errors. We refine the hierarchical chunk-to-string model into two models: a loose model (more similar to the hierarchical phrase-based model) and a tight model (more similar to the tree-to-string model).

Our experiments show that our decoder can improve translation performance significantly over the hierarchical phrase-based decoder and the tree-to-string decoder. Besides, the loose model gives a better performance while the tight model gives a faster speed.

## 8 Acknowledgements

We would like to thank Trevor Cohn, Shujie Liu, Nan Duan, Lei Cui and Mo Yu for their help, and anonymous reviewers for their valuable comments and suggestions. This work was supported in part by EPSRC grant EP/I034750/1 and in part by High Technology R&D Program Project No. 2011AA01A207.

## References

- Colin Cherry. 2008. Cohesive phrase-based decoding for statistical machine translation. In *Proc. of ACL*, pages 72–80.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of ACL*, pages 263–270.
- David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33:201–228.
- Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proc. of ACL*, pages 531–540.
- Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. of ACL*, pages 205–208.
- Yang Feng, Haitao Mi, Yang Liu, and Qun Liu. 2010. An efficient shift-reduce decoding algorithm for phrased-based machine translation. In *Proc. of Coling:Posters*, pages 285–293.
- Heidi Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proc. of EMNLP*, pages 304–311.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a translation rule? In *Proc of NAACL*, pages 273–280.
- Jonathan Graehl and Kevin Knight. 2004. Training tree transducers. In *Proc. of HLT-NAACL*, pages 105–112.
- J Hammersley and P Clifford. 1971. Markov fields on finite graphs and lattices. In *Unpublished manuscript*.
- Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA*.
- Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 127–133.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*, pages 282–289.
- Yang Liu and Qun Liu. 2010. Joint parsing and translation. In *Proc. of COLING*, pages 707–715.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. of COLING-ACL*, pages 609–616.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19:313–330.
- Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proc. of ACL*, pages 1003–1011.
- Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. of ACL*, pages 192–199.
- Thai Phuong Nguyen, Akira Shimazu, Tu Bao Ho, Minh Le Nguyen, and Vinh Van Nguyen. 2008. A tree-to-string phrase-based model for statistical machine translation. In *Proc. of CoNLL*, pages 143–150.
- Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proc. of ACL*.
- Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of ACL*, pages 295–302.
- Frans J. Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30:417–449.
- Frans J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL*, 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*, pages 311–318.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of ACL*, pages 271–279.
- Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proc. of HLT-NAACL*, pages 134–141.
- Ben Taskar, Pieter Abbeel, and Daphne Koller. 2002. Discriminative probabilistic models for relational data. In *Eighteenth Conference on Uncertainty in Artificial Intelligence*.
- Taro Watanabe, Eiichiro Sumita, and Hiroshi G. Okuno. 2003. Chunk-based statistical translation. In *Proc. of ACL*, pages 303–310.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23:377–403.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL*, pages 523–530.

# Large-Scale Syntactic Language Modeling with Treelets

Adam Pauls    Dan Klein  
Computer Science Division  
University of California, Berkeley  
Berkeley, CA 94720, USA  
{adpauls, klein}@cs.berkeley.edu

## Abstract

We propose a simple generative, syntactic language model that conditions on overlapping windows of tree context (or *treelets*) in the same way that  $n$ -gram language models condition on overlapping windows of linear context. We estimate the parameters of our model by collecting counts from automatically parsed text using standard  $n$ -gram language model estimation techniques, allowing us to train a model on over one billion tokens of data using a single machine in a matter of hours. We evaluate on perplexity and a range of grammaticality tasks, and find that we perform as well or better than  $n$ -gram models and other generative baselines. Our model even competes with state-of-the-art discriminative models hand-designed for the grammaticality tasks, despite training on positive data alone. We also show fluency improvements in a preliminary machine translation experiment.

## 1 Introduction

$N$ -gram language models are a central component of all speech recognition and machine translation systems, and a great deal of research centers around refining models (Chen and Goodman, 1998), efficient storage (Pauls and Klein, 2011; Heafield, 2011), and integration into decoders (Koehn, 2004; Chiang, 2005). At the same time, because  $n$ -gram language models only condition on a local window of linear word-level context, they are poor models of long-range syntactic dependencies. Although several lines of work have proposed generative syntactic language models that improve on  $n$ -gram models for moderate amounts of data (Chelba, 1997; Xu et al., 2002; Charniak, 2001; Hall, 2004; Roark,

2004), these models have only recently been scaled to the impressive amounts of data routinely used by  $n$ -gram language models (Tan et al., 2011).

In this paper, we describe a generative, syntactic language model that conditions on local context *treelets*<sup>1</sup> in a parse tree, backing off to smaller *treelets* as necessary. Our model can be trained simply by collecting counts and using the same smoothing techniques normally applied to  $n$ -gram models (Kneser and Ney, 1995), enabling us to apply techniques developed for scaling  $n$ -gram models out of the box (Brants et al., 2007; Pauls and Klein, 2011). The simplicity of our training procedure allows us to train a model on a billion tokens of data in a matter of hours on a single machine, which compares favorably to the more involved training algorithm of Tan et al. (2011), who use a two-pass EM training algorithm that takes several days on several hundred CPUs using similar amounts of data.

The simplicity of our approach also contrasts with recent work on language modeling with tree substitution grammars (Post and Gildea, 2009), where larger *treelet* contexts are incorporated by using sophisticated priors to learn a segmentation of parse trees. Such an approach implicitly assumes that a “correct” segmentation exists, but it is not clear that this is true in practice. Instead, we build upon the success of  $n$ -gram language models, which do not assume a segmentation and instead score all overlapping contexts.

We evaluate our model in terms of perplexity, and show that we achieve the same performance as a state-of-the-art  $n$ -gram model. We also evaluate our model on several grammaticality tasks proposed in

---

<sup>1</sup>We borrow the term *treelet* from Quirk et al. (2005), who use it to refer to an arbitrary connected subgraph of a tree.

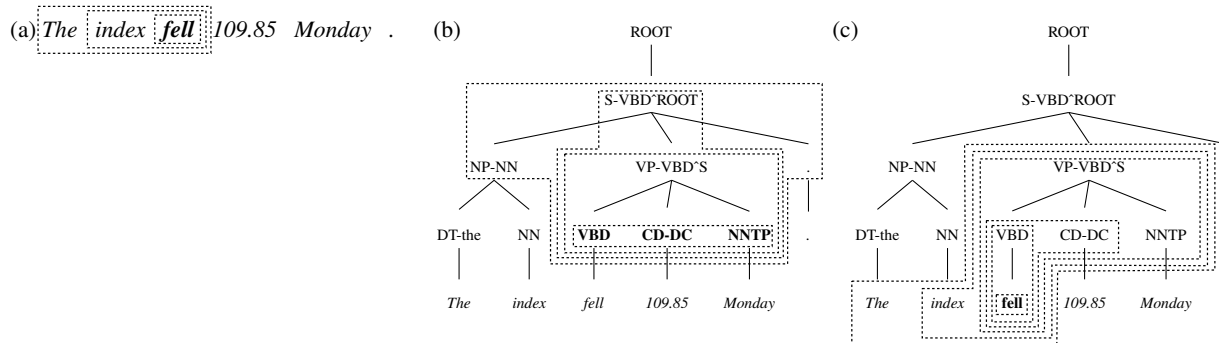


Figure 1: Conditioning contexts and back-off strategies for Markov models. The bolded symbol indicates the part of the tree/sentence being generated, and the dotted lines represent the conditioning contexts; back-off proceeds from the largest to the smallest context. (a) A trigram model. (b) The context used for non-terminal productions in our treelet model. For this context,  $P=VP-VBD^{\wedge}S$ ,  $P'=S-VBD^{\wedge}ROOT$ , and  $r'=S-VBD^{\wedge}ROOT \rightarrow NP-NN VP-VBD^{\wedge}S$ . (c) The context used for terminal productions in our treelet model. Here,  $P=VBD$ ,  $R=CD-DC$ ,  $r'=VP-VBD^{\wedge}S \rightarrow VBD CD-DC NNTP$ ,  $w_{-1}=index$ , and  $w_{-2}=The$ . Note that the tree is a modified version of a standard Penn Treebank parse – see Section 3 for details.

the literature (Okanojima and Tsujii, 2007; Foster et al., 2008; Cherry and Quirk, 2008) and show that it consistently outperforms an  $n$ -gram model as well as other head-driven and tree-driven generative baselines. Our model even competes with state-of-the-art discriminative classifiers specifically designed for each task, despite being estimated on positive data alone. We also show fluency improvements in a preliminary machine translation reranking experiment.

## 2 Treelet Language Modeling

The common denominator of most  $n$ -gram language models is that they assign probabilities roughly according to empirical frequencies for observed  $n$ -grams, but fall back to distributions conditioned on smaller contexts for unobserved  $n$ -grams, as shown in Figure 1(a). This type of smoothing is both highly robust and easy to implement, requiring only the collection of counts from data.

We would like to apply the same smoothing techniques to distributions over rule yields in a constituency tree, conditioned on contexts consisting of previously generated treelets (rules, nodes, etc.). Formally, let  $T$  be a constituency tree consisting of context-free rules of the form  $r = P \rightarrow C_1 \cdots C_d$ , where  $P$  is the parent symbol of rule  $r$  and  $C_1^d = C_1 \cdots C_d$  are its children. We wish to assign probabilities to trees<sup>2</sup>

<sup>2</sup>A distribution over trees also induces a distribution over sentences  $w_1^t$  given by  $p(w_1^t) = \sum_{T:s(T)=w_1^t} p(T)$ , where

$$p(T) = \prod_{r \in T} p(C_1^d | h)$$

where the conditioning context  $h$  is some portion of the already-generated parts of the tree. In this paper, we assume that the children of a rule are expanded from left to right, so that when generating the yield  $C_1^d$ , all treelets above and left of the parent  $P$  are available. Note that a raw PCFG would condition only on  $P$ , i.e.  $h = P$ .

As in the  $n$ -gram case, we would like to pick  $h$  to be large enough to capture relevant dependencies, but small enough that we can obtain meaningful estimates from data. We start with a straightforward choice of context: we condition on  $P$ , as well as the rule  $r'$  that generated  $P$ , as shown in in Figure 1(b).

Conditioning on the parent rule  $r'$  allows us to capture several important dependencies. First, it captures both  $P$  and its parent  $P'$ , which predicts the distribution over child symbols far better than just  $P$  (Johnson, 1998). Second, it captures positional effects. For example, subject and object noun phrases (NPs) have different distributions (Klein and Manning, 2003), and the position of an NP relative to a verb is a good indicator of this distinction. Finally, the generation of words at preterminals can condition on siblings, allowing the model to capture, for example, verb subcategorization frames.

We should be clear that we are not the first  $s(T)$  is the terminal yield of  $T$ .

to use back-off-based smoothing for syntactic language modeling – such techniques have been applied to models that condition on head-word contexts (Charniak, 2001; Roark, 2004; Zhang, 2009). Parent rule context has also been employed in translation (Vaswani et al., 2011). However, to our knowledge, we are the first to apply these techniques for language modeling on large amounts of data.

## 2.1 Lexical context

Although it is tempting to think that we can replace the left-to-right generation of  $n$ -gram models with the purely top-down generation of typical PCFGs, in practice, words are often highly predictive of the words that follow them – indeed,  $n$ -gram models would be terrible language models if this were not the case. To capture linear effects, we extend the context for terminal (lexical) productions to include the previous two words  $w_{-2}$  and  $w_{-1}$  in the sentence in addition to  $r'$ ; see Figure 1(c) for a depiction. This allows us to capture collocations and other lexical correlations.

## 2.2 Backing off

As with  $n$ -gram models, counts for rule yields conditioned on  $r'$  are sparse, and we must choose an appropriate back-off strategy. We handle terminal and non-terminal productions slightly differently.

For non-terminal productions, we back off from  $r'$  to  $P$  and its parent  $P'$ , and then to just  $P$ . That is, we back off from a rule-annotated grammar  $p(C_1^d|P, P', r')$  to a parent-annotated grammar (Johnson, 1998)  $p(C_1^d|P, P')$ , then to a raw PCFG  $p(C_1^d|P)$ . In order to generalize to unseen rule yields  $C_1^d$ , we further back off from the basic PCFG probability  $p(C_1^d|P)$  to  $p(C_i|C_{i-3}^{i-1}, P)$ , a 4-gram model over symbols  $C$  conditioned on  $P$ , interpolated with an unconditional 4-gram model  $p(C_i|C_{i-3}^{i-1})$ . In other words, we back off from a raw PCFG to

$$\lambda \prod_{i=1}^d p(C_i|C_{i-3}^{i-1}, P) + (1 - \lambda) \prod_{i=1}^d p(C_i|C_{i-3}^{i-1})$$

where  $\lambda = 0.9$  is an interpolation constant.

For terminal (i.e. lexical) productions, we first remove lexical context, backing off from

$p(w|P, R, r', w_{-1}, w_{-2})$  to  $p(w|P, R, r', w_{-1})$  and then  $p(w|P, R, r')$ . From there, we back off to  $p(w|P, R)$  where  $R$  is the sibling immediately to the right of  $P$ , then to a raw PCFG  $p(w|P)$ , and finally to a unigram distribution. We chose this scheme because  $p(w|P, R)$  allows, for example, a verb to be generated conditioned on the non-terminal category of the argument it takes (since arguments usually immediately follow verbs). We depict these two back-off schemes pictorially in Figure 1(b) and (c).

## 2.3 Estimation

Estimating the probabilities in our model can be done very simply using the same techniques (in fact, the same code) used to estimate  $n$ -gram language models. Our model requires estimates of four distributions:  $p(C_1^d|P, P', r')$ ,  $p(w|P, R, r', w_{-1}, w_{-2})$ ,  $p(C_i|C_{i-n+1}^{i-1}, P)$ , and  $p(C_i|C_{i-n+1}^{i-1})$ . In each case, we require empirical counts of treelet tuples in the same way that we require counts of word tuples for estimating  $n$ -gram language models.

There is one additional hurdle in the estimation of our model: while there exist corpora with human-annotated constituency parses like the Penn Treebank (Marcus et al., 1993), these corpora are quite small – on the order of millions of tokens – and we cannot gather nearly as many counts as we can for  $n$ -grams, for which billions or even trillions (Brants et al., 2007) of tokens are available on the Web. However, we can use one of several high-quality constituency parsers (Collins, 1997; Charniak, 2000; Petrov et al., 2006) to automatically generate parses. These parses may contain errors, but not all parsing errors are problematic for our model, since we only care about the sentences generated by our model and not the parses themselves. We show in our experiments that the addition of data with automatic parses does improve the performance of our language models across a range of tasks.

## 3 Tree Transformations

In the previous section, we described how to condition on rich parse context to better capture the distribution of English trees. While such context allows our model to capture many interesting dependencies, several important dependencies require additional attention. In this section, we describe a

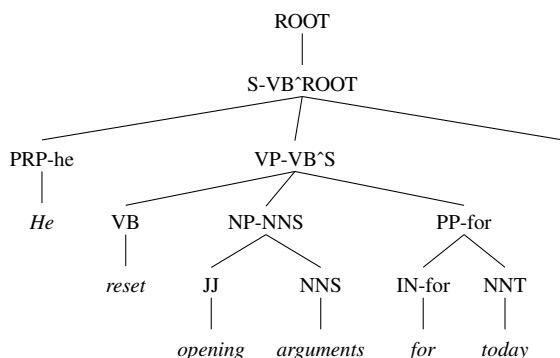


Figure 2: A sample parse from the Penn Treebank after the tree transformations described in Section 3. Note that we have not shown head tag annotations on preterminals because in that case, the head tag is the preterminal itself.

number of transformations of Treebank constituency parses that allow us to capture such dependencies. We list the annotations and deletions in the order in which they are performed. A sample transformed tree is shown in Figure 2.

**Temporal NPs** Following Klein and Manning (2003), we attempt to annotate temporal noun phrases. Although the Penn Treebank annotates temporal NPs, most off-the-shelf parsers do not retain these tags, and we do not assume their presence. Instead, we mark any noun that is the head of a NP-TMP constituent at least once in the Treebank as a temporal noun, so for example *today* would be tagged as NNT and *months* would be tagged as NNTS.

**Head Annotations** We annotate every non-terminal or preterminal with its head word if the head is a closed-class word<sup>3</sup> and with its head tag otherwise. Klein and Manning (2003) used head tag annotation extensively, though they applied their splits much more selectively.

**NP Flattening** We delete NPs dominated by other NPs, unless the child NPs are in coordination or apposition. These NPs typically occur when nouns are modified by PPs, as in (NP (NP (NN stock) (NNS sales)) (PP (IN by) (NNS traders))). By removing the dominated NP, we allow the production  $NNS \rightarrow sales$  to condition on the presence of a modifying PP (here a PP head-annotated with *by*).

**Number Annotations** Numbers are divided into five classes: CD-YR for numbers that consist of four digits (which are usually years); CD-NM for entirely numeric numbers; CD-DC for numbers that have a decimal; CD-

<sup>3</sup>We define the following to be closed class words: any punctuation; all inflections of the verbs *do*, *be*, and *have*; and any word tagged with IN, WDT, PDT, WP, WP\$, TO, WRB, RP, DT, SYM, EX, POS, PRP, AUX, or CC.

MX for numbers that mix letters and digits; and CD-AL for numbers that are entirely alphabetic.

**SBAR Flattening** We remove any sentential (S) nodes immediately dominated by an SBAR. S nodes under SBAR have very distinct distributions from other sentential nodes, mostly due to empty subjects and/or objects.

**VP Flattening** We remove any VPs immediately dominating a VP, unless it is conjoined with another VP. In the Treebank, chains of verbs (e.g. *will be going*) have a separate VP for each verb. By flattening such structures, we allow the main verb and its arguments to condition on the whole chain of verbs. This effect is particularly important for passive constructions.

**Gapped Sentence Annotation** Collins (1999) and Klein and Manning (2003) annotate nodes which have empty subjects. Because we only assume the presence of automatically derived parses, which do not produce the empty elements in the original Treebank, we must identify such elements on our own. We use a very simple procedure: we annotate all S or SBAR nodes that have a VP before any NPs.

**Parent Annotation** We annotate all VPs with their parent symbol. Because our treelet model already conditions on the parent, this has the effect of allowing verbs to condition on their grandparents. This was important for VPs under SBAR nodes, which often have empty objects. We also parent-annotated any child of the ROOT.

**Unary Deletion** We remove all unary productions except the root and preterminal productions, keeping only the bottom-most symbol. Because we are not interested in the internal labels of the trees, unaries are largely a nuisance, and their removal brings many symbols into the context of others.

## 4 Scoring a Sentence

Computing the probability of a sentence  $w_1^\ell$  under our model requires summing over all possible parses of  $w_1^\ell$ . Although our model can be formulated as a straightforward PCFG, allowing  $O(\ell^3)$  computation of this sum, the grammar constant for this PCFG would be unmanageably large (since every parent rule context would need its own state in the grammar), and extensive pruning would be in order.

In practice, however, language models are normally integrated into a decoder, a non-trivial task that is highly problem-dependent and beyond the scope of this paper. For machine translation, a model that builds target-side constituency parses, such as that of Galley et al. (2006), combined with an efficient pruning strategy like cube pruning (Chiang,

5-GRAM
<i>The board 's will soon be feasible , from everyday which Coke 's cabinet hotels . They are all priced became regulatory action by difficulty caused nor Aug. 31 of Helmsley-Spear : Lakeland , it may take them if the 46-year-old said the loss of the Japanese executives at him : But 8.32 % stake in and Rep. any money for you got from several months , " he says .</i>
TREELET
<i>Why a \$ 1.2 million investment in various types of the bulk of TVS E. August ? " One operating price position has a system that has Quartet for the first time , " he said . He may enable drops to take , but will hardly revive the rush to develop two-stroke calculations . " The centers are losses of meals , and the runs are willing to like them .</i>

Table 1: The first four samples of length between 15 and 20 generated from the 5-GRAM and TREELET models.

2005), should be able to integrate our model without much difficulty.

That said, for evaluation purposes, whenever we need to query our model, we use the simple strategy of parsing a sentence using a black box parser, and summing over our model’s probabilities of the 1000-best parses.<sup>4</sup> Note that the bottleneck in this case is the parser, so our model can essentially score a sentence at the speed of a parser.

## 5 Experiments

We evaluate our model along several dimensions. We first show some sample generated sentences in Section 5.1. We report perplexity results in Section 5.2. In Section 5.3, we measure its ability to distinguish between grammatical English and various types of automatically generated, or *pseudo-negative*,<sup>5</sup> English. We report machine translation reranking results in Section 5.4.

### 5.1 Generating Samples

Because our model is generative, we can qualitatively assess it by generating samples and verifying that they are more syntactically coherent than other approaches. In Table 1, we show the first four samples of length between 15 and 20 generated from our model and a 5-gram model trained on the Penn Treebank.

<sup>4</sup>We found that using the 1-best worked just as well as the 1000-best on our grammaticality tasks, but significantly overestimated our model’s perplexities.

<sup>5</sup>We follow Okanojima and Tsujii (2007) in using the term pseudo-negative to highlight the fact that automatically generated negative examples might not actually be ungrammatical.

### 5.2 Perplexity

Perplexity is the standard intrinsic evaluation metric for language models. It measures the inverse of the per-word probability a model assigns to some held-out set of grammatical English (so lower is better). For training data, we constructed a large treebank by concatenating the WSJ and Brown portions of the Penn Treebank, the 50K BLLIP training sentences from Post (2011), and the AFP and APW portions of English Gigaword version 3 (Graff, 2003), totaling about 1.3 billion tokens. We used the human-annotated parses for the sentences in the Penn Treebank, but parsed the Gigaword and BLLIP sentences with the Berkeley Parser. Hereafter, we refer to this training data as our 1B corpus. We used Section 0 of the WSJ as our test corpus. Results are shown in Table 2. In addition to our TREELET model, we also show results for the following baselines:

**5-GRAM** A 5-gram interpolated Kneser-Ney model.

**PCFG-LA** The Berkeley Parser in language model mode.

**HEADLEX** A head-lexicalized model similar to, but more powerful<sup>6</sup> than, Collins Model 1 (Collins, 1999).

**PCFG** A raw PCFG.

**TREELET-TRANS** A PCFG estimated on the trees after the transformations of Section 3.

**TREELET-RULE** The TREELET-TRANS model with the parent rule context described in Section 2. This is equivalent to the full TREELET model without the lexical context described in Section 2.1.

<sup>6</sup>Specifically, like Collins Model 1, we generate a rule yield conditioned on parent symbol  $P$  and head word  $h$  by first generating its head symbol  $C_h$ , then generating the head words and symbols for left and right modifiers outwards from  $C_h$ . Unlike Model 1, which generates each modifier head and symbol conditioned only on  $C_h$ ,  $h$ , and  $P$ , we additionally condition on the previously generated modifier’s head and symbol and back off to Model 1.

Model	Perplexity
PCFG	1772
TREELET-TRANS	722
TREELET-RULE	329
TREELET	<b>198<sup>†</sup></b>
PCFG-LA	330**
HEADLEX	299
5-GRAM	207 <sup>†</sup>

Table 2: Perplexity of several generative models on Section 0 of the WSJ. The differences between scores marked with <sup>†</sup> are not statistically significant. PCFG-LA (marked with \*\*) was only trained on the WSJ and Brown corpora because it does not scale to large amounts of data.

We used the Berkeley LM toolkit (Pauls and Klein, 2011), which implements Kneser-Ney smoothing, to estimate all back-off models for both  $n$ -gram and treelet models. To deal with unknown words, we use the following strategy: after the first 10000 sentences, whenever we see a new word in our training data, we replace it with a signature<sup>7</sup> 10% of the time.

Our model outperforms all other generative models, though the improvement over the  $n$ -gram model is not statistically significant. Note that because we use a  $k$ -best approximation for the sum over trees, all perplexities (except for PCFG-LA and 5-GRAM) are pessimistic bounds.

### 5.3 Classification of Pseudo-Negative Sentences

We make use of three kinds of automatically generated pseudo-negative sentences previously proposed in the literature: Okanohara and Tsujii (2007) proposed generating pseudo-negative examples from a trigram language model; Foster et al. (2008) create “noisy” sentences by automatically inserting a single error into grammatical sentences with a script that randomly deletes, inserts, or misspells a word; and Och et al. (2004) and Cherry and Quirk (2008) both use the 1-best output of a machine translation system. Examples of these three types of pseudo-negative data are shown in Table 3. We evaluate our model’s ability to distinguish positive from pseudo-negative data, and compare against generative baselines and state-of-the-art discriminative methods.

<sup>7</sup>We use signatures generated by the Berkeley Parser. These signatures capture surface features such as capitalization, presents of digits, and common suffixes. For example, the word *vexing* would be replaced with the signature UNK-ing.

<b>Noisy</b>	<i>There was were many contributors.</i>
<b>Trigram</b>	<i>For years in dealer immediately .</i>
<b>MT</b>	<i>we must further steps .</i>

Table 3: Sample pseudo-negative sentences.

We would like to use our model to make grammaticality judgements, but as a generative model it can only provide us with probabilities. Simply thresholding generative probabilities, even with a separate threshold for each length, has been shown to be very ineffective for grammaticality judgements, both for  $n$ -gram and syntactic language models (Cherry and Quirk, 2008; Post, 2011). We used a simple measure for isolating the syntactic likelihood of a sentence: we take the log-probability under our model and subtract the log-probability under a unigram model, then normalize by the length of the sentence.<sup>8</sup> This measure, which we call the *syntactic log-odds ratio* (SLR), is a crude way of “subtracting out” the semantic component of the generative probability, so that sentences that use rare words are not penalized for doing so.

#### 5.3.1 Trigram Classification

To facilitate comparison with previous work, we used the same negative corpora as Post (2011) for trigram classification. They randomly selected 50K train, 3K development, and 3K positive test sentences from the BLLIP corpus, then trained a trigram model on 450K BLLIP sentences and generated 50K train, 3K development, and 3K negative sentences. We parsed the 50K positive training examples of Post (2011) with the Berkeley Parser and used the resulting treebank to train a treelet language model. We set an SLR threshold for each model on the 6K positive and negative development sentences.

Results are shown in Table 4. In addition to our generative baselines, we show results for the discriminative models reported in Cherry and Quirk (2008) and Post (2011). The former train a latent PCFG support vector machine for binary classification (LSVM). The latter report results for two binary classifiers: RERANK uses the reranking features of Charniak and Johnson (2005), and TSG uses

<sup>8</sup>Och et al. (2004) also report using a parser probability normalized by the unigram probability (but not length), and did not find it effective. We assume this is either because the length-normalization is important, or because their choice of syntactic language model was poor.



Generative		
	BLLIP	1B
PCFG	81.5	81.8
TREELET-TRANS	87.7	90.1
TREELET-RULE	89.8	<b>94.1</b>
TREELET	88.9	93.3
PCFG-LA	87.1*	–
HEADLEX	87.6	92.0
5-GRAM	67.9	87.5
Discriminative		
	BLLIP	1B
LSVM	81.42**	–
TSG	89.9	–
RERANK	<b>93.0</b>	–

Table 4: Classification accuracy for trigram pseudo-negative sentences on the BLLIP corpus. The number reported for PCFG-LA is marked with a \* to indicate that this model was trained on the training section of the WSJ, not the BLLIP corpus. The number reported for LSVM (marked with \*\*) was evaluated on a different random split of the BLLIP corpus, and so is not directly comparable.

indicator features extracted from a tree substitution grammar derivation of each sentence.

Our TREELET model performs nearly as well as the TSG method, and substantially outperforms the LSVM method, though the latter was not tested on the same random split. Interestingly, the TREELET-RULE baseline, which removes lexical context from our model, outperforms the full model. This is likely because the negative data is largely coherent at the trigram level (because it was generated from a trigram model), and the full model is much more sensitive to trigram coherence than the TREELET-RULE model. This also explains the poor performance of the 5-GRAM model.

We emphasize that the discriminative baselines are *specifically* trained to separate trigram text from natural English, while our model is trained on positive examples alone. Indeed, the methods in Post (2011) are simple binary classifiers, and it is not clear that these models would be properly calibrated for any other task, such as integration in a decoder.

One of the design goals of our system was that it be scalable. Unlike some of the discriminative baselines, which require expensive operations<sup>9</sup> on

<sup>9</sup>It is true that in order to train our system, one must parse large amounts of training data, which can be costly, though it only needs to be done once. In contrast, even with observed training trees, the discriminative algorithms must still iteratively perform expensive operations (like parsing) for each sentence, and a new model must be trained for new types of negative data.

Model	Pairwise		Independent	
	WSJ	1B	WSJ	1B
PCFG	79.1	77.0	58.9	58.6
TREELET-RULE	90.3	94.4	63.8	<b>66.2</b>
TREELET	90.7	<b>94.5</b>	63.4	65.5
5-GRAM	86.3	93.5	55.7	60.1
HEADLEX	90.7	94.0	59.5	62.0
PCFG-LA	<b>91.3</b>	–	59.7	–
Foster et al. (2008)	–	–	<b>65.9</b>	–

Table 5: Classification accuracies on the noisy WSJ for models trained on WSJ Sections 2-21 and our 1B token corpus. “Pairwise” accuracy is the fraction of correct sentences whose SLR score was higher than its noisy version, and “independent” refers to standard binary classification accuracy.

each training sentence, we can very easily scale our model to much larger amounts of data. In Table 4, we also show the performance of the generative models trained on our 1B corpus. All generative models improve, but TREELET-RULE remains the best, now outperforming the RERANK system, though of course it is likely that RERANK would improve if it could be scaled up to more training data.

### 5.3.2 “Noisy” Classification

We also evaluate the performance of our model on the task of distinguishing the noisy WSJ sentences of Foster et al. (2008) from their original versions. We use the noisy versions of Section 0 and 23 produced by their error-generating procedure. Because they only report classification results on Section 0, we used Section 23 to tune an SLR threshold, and tested our model on Section 0. We show the results of both independent and pairwise classification for the WSJ and 1B training sets in Table 5. Note that independent classification is much more difficult than for the trigram data, because sentences contain at most one change, which may not even result in an ungrammaticality. Again, our model outperforms the  $n$ -gram model for both types of classification, and achieves the same performance as the discriminative system of Foster et al. (2008), which is state-of-the-art for this data set. The TREELET-RULE system again slightly outperforms the full TREELET model at independent classification, though not at pairwise classification. This probably reflects the fact that semantic coherence can still influence the SLR score, despite our efforts to subtract it out. Because the TREELET model includes lexical context, it is more sensitive to seman-

	French	German	Chinese
5-GRAM	44.8	37.8	60.0
TREELET	<b>57.9</b>	<b>66.0</b>	<b>83.8</b>

Table 6: Pairwise comparison accuracy of MT output against a reference translation for French, German, and Chinese. The BLEU scores for these outputs are 32.7, 27.8, and 20.8. This task becomes easier, at least for our TREELET model, as translation quality drops. Cherry and Quirk (2008) report an accuracy of 71.9% on a similar experiment with German a source language, though the translation system and training data were different so the numbers are not comparable. In particular, their translations had a lower BLEU score, making their task easier.

tic coherence and thus more likely to misclassify semantically coherent but ungrammatical sentences. For pairwise comparisons, where semantic coherence is effectively held constant, such sentences are not problematic.

### 5.3.3 Machine Translation Classification

We follow Och et al. (2004) and Cherry and Quirk (2008) in evaluating our language models on their ability to distinguish the 1-best output of a machine translation system from a reference translation in a pairwise fashion. Unfortunately, we do not have access to the data used in those papers, so a direct comparison is not possible. Instead, we collected the English output of Moses (Hoang et al., 2007), using both French and German as source language, trained on the Europarl corpus used by WMT 2009.<sup>10</sup> We also collected the output of Joshua (Li et al., 2009) trained on 500K sentences of GALE Chinese-English parallel newswire. We trained both our TREELET model and a 5-GRAM model on the union of our 1B corpus and the English sides of our parallel corpora.

In Table 6, we show the pairwise comparison accuracy (using SLR) on these three corpora. We see that our system prefers the reference much more often than the 5-GRAM language model.<sup>11</sup> However, we also note that the easiness of the task is correlated with the quality of translations (as measured in BLEU score). This is not surprising – high-quality translations are often grammatical and even a per-

<sup>10</sup><http://www.statmt.org/wmt09>

<sup>11</sup>We note that the  $n$ -gram language model used by the MT system was much smaller than the 5-GRAM model, as they were only trained on the English sides of their parallel data.

fect language model might not be able to differentiate such translations from their references.

### 5.4 Machine Translation Fluency

We also carried out reranking experiments on 1000-best lists from Moses using our syntactic language model as a feature. We did not find that the use of our syntactic language model made any statistically significant increases in BLEU score. However, we noticed in general that the translations favored by our model were more fluent, a useful improvement to which BLEU is often insensitive. To confirm this, we carried out an Amazon Mechanical Turk experiment where users from the United States were asked to compare translations using our TREELET language model as the language model feature to those using the 5-GRAM model.<sup>12</sup> We had 1000 such translation pairs rated by 4 separate Turkers each. Although these two hypothesis sets had the same BLEU score (up to statistical significance), the Turkers preferred the output obtained using our syntactic language model 59% of the time, indicating that our model had managed to pick out more fluent hypotheses that nonetheless were of the same BLEU score. This result was statistically significant with  $p < 0.001$  using bootstrap resampling.

## 6 Conclusion

We have presented a simple syntactic language model that can be estimated using standard  $n$ -gram smoothing techniques on large amounts of data. Our model outperforms generative baselines on several evaluation metrics and achieves the same performance as state-of-the-art discriminative classifiers specifically trained on several types of negative data.

### Acknowledgments

We would like to thank David Hall for some modeling suggestions and the anonymous reviewers for their comments. We thank both Matt Post and Jennifer Foster for providing us with their corpora. This work was partially supported by a Google Fellowship to the first author and by BBN under DARPA contract HR0011-12-C-0014.

<sup>12</sup>We used translations from the baseline Moses system of Section 5.3.3 with German as the input language. For each language model, we took  $k$ -best lists from the baseline system and replaced the baseline LM score with the new model’s score. We then retrained all feature weights with MERT on the tune set, and selected the 1-best output on the test set.

## References

- Thorsten Brants, Ashok C. Popat, Peng Xu, Franz J. Och, Jeffrey Dean, and Google Inc. 2007. Large language models in machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the Association for Computational Linguistics*.
- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the North American chapter of the Association for Computational Linguistics*.
- Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the Association for Computational Linguistics*.
- Ciprian Chelba. 1997. A structured language model. In *Proceedings of the Association for Computational Linguistics*.
- Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. In *Proceedings of the Association for Computational Linguistics*.
- Colin Cherry and Chris Quirk. 2008. Discriminative, syntactic language modeling through latent SVMs. In *Proceedings of The Association for Machine Translation in the Americas*.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *The Annual Conference of the Association for Computational Linguistics*.
- Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of Association for Computational Linguistics*.
- Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.
- Jennifer Foster, Joachim Wagner, and Josef van Genabith. 2008. Adapting a wsj-trained parser to grammatically noisy text. In *Proceedings of the Association for Computational Linguistics: Short Paper Track*.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *The Annual Conference of the Association for Computational Linguistics (ACL)*.
- David Graff. 2003. English gigaword, version 3. In *Linguistic Data Consortium, Philadelphia, Catalog Number LDC2003T05*.
- Keith Hall. 2004. *Best-first Word-lattice Parsing: Techniques for Integrated Syntactic Language Modeling*. Ph.D. thesis, Brown University.
- Kenneth Heafield. 2011. Kenlm: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*.
- Hieu Hoang, Alexandra Birch, Chris Callison-burch, Richard Zens, Rwth Aachen, Alexandra Constantin, Marcello Federico, Nicola Bertoldi, Chris Dyer, Brooke Cowan, Wade Shen, Christine Moran, and Ondrej Bojar. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Association for Computational Linguistics: Demonstration Session*.
- Mark Johnson. 1998. PCFG models of linguistic tree representations. *Computational Linguistics*, 24.
- Dan Klein and Chris Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.
- Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *IEEE International Conference on Acoustics, Speech and Signal Processing*.
- Philipp Koehn. 2004. Pharaoh: A beam search decoder for phrase-based statistical machine translation models. In *Proceedings of The Association for Machine Translation in the Americas*.
- Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren N. G. Thornton, Jonathan Weese, and Omar F. Zaidan. 2009. Joshua: an open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*.
- M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. In *Computational Linguistics*.
- Franz J. 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. In *Proceedings of the North American Association for Computational Linguistic*.
- Daisuke Okanohara and Jun'ichi Tsujii. 2007. A discriminative language model with pseudo-negative samples. In *Proceedings of the Association for Computational Linguistics*.
- Adam Pauls and Dan Klein. 2011. Faster and smaller  $n$ -gram language models. In *Proceedings of the Association for Computational Linguistics*.
- Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of COLING-ACL 2006*.
- Matt Post and Daniel Gildea. 2009. Language modeling with tree substitution grammars. In *Proceedings*

*of the Conference on Neural Information Processing Systems.*

- Matt Post. 2011. Judging grammaticality with tree substitution grammar. In *Proceedings of the Association for Computational Linguistics: Short Paper Track*.
- Chris Quirk, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of the Association of Computational Linguistics*.
- Brian Roark. 2004. Probabilistic top-down parsing and language modeling. *Computational Linguistics*.
- Ming Tan, Wenli Zhou, Lei Zheng, and Shaojun Wang. 2011. A large scale distributed syntactic, semantic and lexical language model for machine translation. In *Proceedings of the Association for Computational Linguistics*.
- Ashish Vaswani, Haitao Mi, Liang Huang, and David Chiang. 2011. Rule markov models for fast tree-to-string translation. In *Proceedings of the Association for Computations Linguistics*.
- Peng Xu, Ciprian Chelba, and Fred Jelinek. 2002. A study on richer syntactic dependencies for structured language modeling. In *Proceedings of the Association for Computational Linguistics*. Association for Computational Linguistics.
- Ying Zhang. 2009. *Structured language models for statistical machine translation*. Ph.D. thesis, Johns Hopkins University.

# Text Segmentation by Language Using Minimum Description Length

**Hiroshi Yamaguchi**

Graduate School of  
Information Science and Technology,  
University of Tokyo

yamaguchi.hiroshi@ci.i.u-tokyo.ac.jp

**Kumiko Tanaka-Ishii**

Faculty and Graduate School of Information  
Science and Electrical Engineering,  
Kyushu University

kumiko@ait.kyushu-u.ac.jp

## Abstract

The problem addressed in this paper is to segment a given multilingual document into segments for each language and then identify the language of each segment. The problem was motivated by an attempt to collect a large amount of linguistic data for non-major languages from the web. The problem is formulated in terms of obtaining the minimum description length of a text, and the proposed solution finds the segments and their languages through dynamic programming. Empirical results demonstrating the potential of this approach are presented for experiments using texts taken from the Universal Declaration of Human Rights and Wikipedia, covering more than 200 languages.

## 1 Introduction

For the purposes of this paper, a multilingual text means one containing text segments, limited to those longer than a clause, written in different languages. We can often find such texts in linguistic resources collected from the World Wide Web for many non-major languages, which tend to also contain portions of text in a major language. In automatic processing of such multilingual texts, they must first be segmented by language, and the language of each segment must be identified, since many state-of-the-art NLP applications are built by learning a gold standard for one specific language. Moreover, segmentation is useful for other objectives such as collecting linguistic resources for non-major languages and automatically removing portions written in major languages, as noted above. The study reported here was motivated by this objective. The problem addressed in this article is thus to segment a multilingual text by language and identify the language of each segment. In addition, for our objective, the set of target languages consists of not only major languages but also many non-major languages: more than 200 languages in total.

Previous work that directly concerns the problem

addressed in this paper is rare. The most similar previous work that we know of comes from two sources and can be summarized as follows. First, (Teahan, 2000) attempted to segment multilingual texts by using text segmentation methods used for non-segmented languages. For this purpose, he used a gold standard of multilingual texts annotated by borders and languages. This segmentation approach is similar to that of word segmentation for non-segmented texts, and he tested it on six different European languages. Although the problem setting is similar to ours, the formulation and solution are different, particularly in that our method uses only a monolingual gold standard, not a multilingual one as in Teahan's study. Second, (Alex, 2005) (Alex et al., 2007) solved the problem of detecting words and phrases in languages other than the principal language of a given text. They used statistical language modeling and heuristics to detect foreign words and tested the case of English embedded in German texts. They also reported that such processing would raise the performance of German parsers. Here again, the problem setting is similar to ours but not exactly the same, since the embedded text portions were assumed to be words. Moreover, the authors only tested for the specific language pair of English embedded in German texts. In contrast, our work considers more than 200 languages, and the portions of embedded text are larger: up to the paragraph level to accommodate the reality of multilingual texts. The extension of our work to address the foreign word detection problem would be an interesting future work.

From a broader view, the problem addressed in this paper is further related to two genres of previous work. The first genre is text segmentation. Our problem can be situated as a sub-problem from the viewpoint of language change. A more common setting in the NLP context is segmentation into semantically coherent text portions, of which a representative method is *text tiling* as reported by (Hearst, 1997). There could be other possible bases for text

segmentation, and our study, in a way, could lead to generalizing the problem. The second genre is classification, and the specific problem of text classification by language has drawn substantial attention (Grefenstette, 1995) (Kruengkrai et al., 2005) (Kikui, 1996). Current state-of-the-art solutions use machine learning methods for languages with abundant supervision, and the performance is usually high enough for practical use. This article concerns that problem together with segmentation but has another particularity in aiming at classification into a substantial number of categories, i.e., more than 200 languages. This means that the amount of training data has to remain small, so the methods to be adopted must take this point into consideration. Among works on text classification into languages, our proposal is based on previous studies using cross-entropy such as (Teahan, 2000) and (Juola, 1997). We explain these works in further detail in §3.

This article presents one way to formulate the segmentation and identification problem as a combinatorial optimization problem; specifically, to find the set of segments and their languages that minimizes the description length of a given multilingual text. In the following, we describe the problem formulation and a solution to the problem, and then discuss the performance of our method.

## 2 Problem Formulation

In our setting, we assume that a small amount (up to kilobytes) of monolingual plain text sample data is available for every language, e.g., the Universal Declaration of Human Rights, which serves to generate the language model used for language identification. This entails two sub-assumptions.

First, we assume that for all multilingual text, every text portion is written in one of the given languages; there is no input text of an unknown language without learning data. In other words, we use supervised learning. In line with recent trends in unsupervised segmentation, the problem of finding segments without supervision could be solved through approaches such as Bayesian methods; however, we report our result for the supervised setting since we believe that every segment must be labeled by language to undergo further processing.

Second, we cannot assume a large amount of

learning data, since our objective requires us to consider segmentation by both major and non-major languages. For most non-major languages, only a limited amount of corpus data is available.<sup>1</sup>

This constraint suggests the difficulty of applying certain state-of-the-art machine learning methods requiring a large learning corpus. Hence, our formulation is based on the minimum description length (MDL), which works with relatively small amounts of learning data.

In this article, we use the following terms and notations. A multilingual text to be segmented is denoted as  $X = x_1, \dots, x_{|X|}$ , where  $x_i$  denotes the  $i$ -th character of  $X$  and  $|X|$  denotes the text's length. *Text segmentation by language* refers here to the process of segmenting  $X$  by a set of borders  $\mathbb{B} = [B_1, \dots, B_{|\mathbb{B}|}]$ , where  $|\mathbb{B}|$  denotes the number of borders, and each  $B_i$  indicates the location of a language border as an offset number of characters from the beginning. Note that a pair of square brackets indicates a list. Segmentation in this paper is character-based, i.e., a  $B_i$  may refer to a position inside a word. The list of *segments* obtained from  $\mathbb{B}$  is denoted as  $\mathbb{X} = [X_0, \dots, X_{|\mathbb{B}|}]$ , where the concatenation of the segments equals  $X$ . The language of each segment  $X_i$  is denoted as  $L_i$ , where  $L_i \in \mathcal{L}$ , the set of languages. Finally,  $\mathbb{L} = [L_0, \dots, L_{|\mathbb{B}|}]$  denotes the sequence of languages corresponding to each segment  $X_i$ . The elements in each adjacent pair in  $\mathbb{L}$  must be different.

We formulate the problem of segmenting a multilingual text by language as follows. Given a multilingual text  $X$ , the segments  $\mathbb{X}$  for a list of borders  $\mathbb{B}$  are obtained with the corresponding languages  $\mathbb{L}$ . Then, the total description length is obtained by calculating each description length of a segment  $X_i$  for the language  $L_i$ :

$$(\hat{\mathbb{X}}, \hat{\mathbb{L}}) = \arg \min_{\mathbb{X}, \mathbb{L}} \sum_{i=0}^{|\mathbb{B}|} dl_{L_i}(X_i). \quad (1)$$

The function  $dl_{L_i}(X_i)$  calculates the description length of a text segment  $X_i$  through the use of a language model for  $L_i$ . Note that the actual total description length must also include an additional term,  $\log_2 |X|$ , giving information on the number of segments (with the maximum to be segmented

<sup>1</sup>In fact, our first motivation was to collect a certain amount of corpus data for non-major languages from Wikipedia.

by each character). Since this term is a common constant for all possible segmentations and the minimization of formula (1) is not affected by this term, we will ignore it.

The model defined by (1) is additive for  $X_i$ , so the following formula can be applied to search for language  $L_i$  given a segment  $X_i$ :

$$\hat{L}_i = \arg \min_{L_i \in \mathcal{L}} dl_{L_i}(X_i), \quad (2)$$

under the constraint that  $L_i \neq L_{i-1}$  for  $i \in \{1, \dots, |\mathbb{B}|\}$ . The function  $dl$  can be further decomposed as follows to give the description length in an information-theoretic manner:

$$dl_{L_i}(X_i) = -\log_2 P_{L_i}(X_i) + \log_2 |X| + \log_2 |\mathcal{L}| + \gamma. \quad (3)$$

Here, the first term corresponds to the code length of the text chunk  $X_i$  given a language model for  $L_i$ , which in fact corresponds to the cross-entropy of  $X_i$  for  $L_i$  multiplied by  $|X_i|$ . The remaining terms give the code lengths of the parameters used to describe the length of the first term: the second term corresponds to the segment location; the third term, to the identified language; and the fourth term, to the language model of language  $L_i$ . This fourth term will differ according to the language model type; moreover, its value can be further minimized through formula (2). Nevertheless, since we use a uniform amount of training data for every language, and since varying  $\gamma$  would prevent us from improving the efficiency of dynamic programming, as explained in §4, in this article we set  $\gamma$  to a constant obtained empirically.

Under this formulation, therefore, when detecting the language of a segment as in formula (2), the terms of formula (3) other than the first term will be constant: what counts is only the first term, similarly to much of the previous work explained in the following section. We thus perform language detection itself by minimizing the cross-entropy rather than the MDL. For segmentation, however, the constant terms function as overhead and also serve to prohibit excessive decomposition.

Next, after briefly introducing methods to calculate the first term of formula (3), we explain the solution to optimize the combinatorial problem of formula (1).

### 3 Calculation of Cross-Entropy

The first term of (3),  $-\log_2 P_{L_i}(X_i)$ , is the cross-entropy of  $X_i$  for  $L_i$  multiplied by  $|X_i|$ . Various methods for computing cross-entropy have been proposed, and these can be roughly classified into two types based on different methods of universal coding and the language model. For example, (Benedetto et al., 2002) and (Cilibrasi and Vitányi, 2005) used the universal coding approach, whereas (Teahan and Harper, 2001) and (Sibun and Reynar, 1996) were based on language modeling using PPM and Kullback-Leibler divergence, respectively.

In this section, we briefly introduce two methods previously studied by (Juola, 1997) and (Teahan, 2000) as representative of the two types, and we further explain a modification that we integrate into the final optimization problem. We tested several other coding methods, but they did not perform as well as these two methods.

#### 3.1 Mean of Matching Statistics

(Farach et al., 1994) proposed a method to estimate the entropy, through a simplified version of the LZ algorithm (Ziv and Lempel, 1977), as follows. Given a text  $X = x_1x_2 \dots x_ix_{i+1} \dots$ ,  $Len_i$  is defined as the longest match length for two substrings  $x_1x_2 \dots x_i$  and  $x_{i+1}x_{i+2} \dots$ . In this article, we define the longest match for two strings A and B as the shortest prefix of string B that is not a substring of A. Letting the average of  $Len_i$  be  $E[Len]$ , Farach proved that  $|E[Len] - \frac{\log_2 i}{H(X)}|$  probabilistically converges to zero as  $i \rightarrow \infty$ , where  $H(X)$  indicates the entropy of  $X$ . Then,  $H(X)$  is estimated as

$$\hat{H}(X) = \frac{\log_2 i}{E[Len]}.$$

(Juola, 1997) applied this method to estimate the cross-entropy of two given texts. For two strings  $Y = y_1y_2 \dots y_{|Y|}$  and  $X = x_1x_2 \dots x_{|X|}$ , let  $Len_i(Y)$  be the match length starting from  $x_i$  of  $X$  for  $Y^2$ . Based on this formulation, the cross-entropy is approximately estimated as

$$\hat{J}_Y(X) = \frac{\log_2 |Y|}{E[Len_i(Y)]}.$$

<sup>2</sup>This is called a *matching statistics* value, which explains the subsection title.

Since formula (1) of §2 is based on adding the description length, it is important that the whole value be additive to enable efficient optimization (as will be explained in §4). We thus modified Juola’s method as follows to make the length additive:

$$\hat{J}'_Y(X) = E \left[ \frac{\log_2 |Y|}{Len_i(Y)} \right].$$

Although there is no mathematical guarantee that  $\hat{J}_Y(X)$  or  $\hat{J}'_Y(X)$  actually converges to the cross-entropy, our empirical tests showed a good estimate for both cases<sup>3</sup>. In this article, we use  $\hat{J}'_Y(X)$  as a function to obtain the cross-entropy and for multiplication by  $|X|$  in formula (3).

### 3.2 PPM

As a representative method for calculating the cross-entropy through statistical language modeling, we adopt prediction by partial matching (PPM), a language-based encoding method devised by (Cleary and Witten, 1984). It has the particular characteristic of using a variable  $n$ -gram length, unlike ordinary  $n$ -gram models<sup>4</sup>. It models the probability of a text  $X$  with a learning corpus  $Y$  as follows:

$$\begin{aligned} P_Y(X) &= P_Y(x_1 \dots x_{|X|}) \\ &= \prod_{t=1}^{|X|} P_Y(x_t | x_{t-1} \dots x_{\max(1, t-n)}), \end{aligned}$$

where  $n$  is a parameter of PPM, denoting the maximum length of the  $n$ -grams considered in the model<sup>5</sup>. The probability  $P_Y(X)$  is estimated by *escape probabilities* favoring the longer sequences appearing in the learning corpus (Bell et al., 1990). The total code length of  $X$  is then estimated as  $-\log P_Y(X)$ . Since this value is additive and gives the total code length of  $X$  for language  $Y$ , we adopt this value in our approach.

## 4 Segmentation by Dynamic Programming

By applying the above methods, we propose a solution to formula (1) through dynamic programming.

<sup>3</sup>This modification means that the original  $\hat{J}_Y(X)$  is obtained through the harmonic mean, with  $Len$  obtained through the arithmetic mean, whereas  $\hat{J}'_Y(X)$  is obtained through the arithmetic mean with  $Len$  as the harmonic mean.

<sup>4</sup>In the context of NLP, this is known as Witten-Bell smoothing.

<sup>5</sup>In the experiments reported here,  $n$  is set to 5 throughout.

Considering the additive characteristic of the description length formulated previously as formula (1), we denote the minimized description length for a given text  $X$  simply as  $DP(X)$ , which can be decomposed recursively as follows<sup>6</sup>:

$$\begin{aligned} DP(X) &= \min_{t \in \{0, \dots, |X|\}, L \in \mathcal{L}} \{ DP(x_0 \dots x_{t-1}) \\ &\quad + dl_L(x_t \dots x_{|X|}) \}, \end{aligned} \quad (4)$$

In other words, the computation of  $DP(X)$  is decomposed into obtaining the addition of two terms by searching through  $t \in \{0, \dots, |X|\}$  and  $L \in \mathcal{L}$ . The first term gives the MDL for the first  $t$  characters of text  $X$ , while the second term,  $dl_L(x_{t+1} \dots x_{|X|})$ , gives the description length of the remaining characters under the language model for  $L$ .

We can straightforwardly implement this recursive computation through dynamic programming, by managing a table of size  $|X| \times |\mathcal{L}|$ . To fill a cell of this table, formula (4) suggests referring to  $t \times |\mathcal{L}|$  cells and calculating the description length of the rest of the text for  $O(|X| - t)$  cells for each language. Since  $t$  ranges up to  $|X|$ , the brute-force computational complexity is  $O(|X|^3 \times |\mathcal{L}|^2)$ .

The complexity can be greatly reduced, however, when the function  $dl$  is additive. First, the description length can be calculated from the previous result, decreasing  $O(|X| - t)$  to  $O(1)$  (to obtain the code length of an additional character). Second, the referred number of cells  $t \times |\mathcal{L}|$  is in fact  $U \times |\mathcal{L}|$ , with  $U \ll |X|$ : for MMS,  $U$  can be proven to be  $O(\log |Y|)$ , where  $|Y|$  is the maximum length among the learning corpora; and for PPM,  $U$  corresponds to the maximum length of an  $n$ -gram. Third, this factor  $U \times |\mathcal{L}|$  can be further decreased to  $U \times 2$ , since it suffices to possess the results for the two<sup>7</sup> best languages in computing the first term of (4). Consequently, the complexity decreases to  $O(U \times |X| \times |\mathcal{L}|)$ .

<sup>6</sup>This formula can be used directly to generate a set  $\mathbb{L}$  in which all adjacent elements differ. The formula can also be used to generate segments for which some adjacent languages coincide and then further to generate  $\mathbb{L}$  through post-processing by concatenating segments of the same language.

<sup>7</sup>This number means the two best scores for different languages, which is required to obtain  $\mathbb{L}$  directly: in addition to the best score, if the language of the best coincides with  $L$  in formula (4), then the second best is also needed. If segments are subjected to post-processing, this value can be one.



Table 1: Number of languages for each writing system

character kinds	UDHR	Wiki
Latin	260	158
Cyrillic	12	20
Devanagari	0	8
Arabic	1	6
Other	4	30

## 5 Experimental Setting

### 5.1 Monolingual Texts (Training / Test Data)

In this work, monolingual texts were used both for training the cross-entropy computation and as test data for cross-validation: the training data does not contain any test data at all. Monolingual texts were also used to build multilingual texts, as explained in the following subsection.

Texts were collected from the World Wide Web and consisted of two sets. The first data set consisted of texts from the Universal Declaration of Human Rights (UDHR)<sup>8</sup>. We consider UDHR the most suitable text source for our purpose, since the content of every monolingual text in the declaration is unique. Moreover, each text has the tendency to maximally use its own language and avoid vocabulary from other languages. Therefore, UDHR-derived results can be considered to provide an empirical upper bound on our formulation. The set  $\mathcal{L}$  consists of 277 languages, and the texts consist of around 10,000 characters on average.

The second data set was Wikipedia data from Wikipedia Downloads<sup>9</sup>, denoted as “Wiki” in the following discussion. We automatically assembled the data through the following steps. First, tags in the Wikipedia texts were removed. Second, short lines were removed since they typically are not sentences. Third, the amount of data was set to 10,000 characters for every language, in correspondence with the size of the UDHR texts. Note that there is a limit to the complete cleansing of data. After these steps, the set  $\mathcal{L}$  contained 222 languages with sufficient data for the experiments.

Many languages adopt writing systems other than the Latin alphabet. The numbers of languages for various representative writing systems are listed in Table 1 for both UDHR and Wiki, while the Ap-

pendix at the end of the article lists the actual languages. Note that in this article, a character means a Unicode character throughout, which differs from a character rendered in block form for some writing systems.

To evaluate language identification for monolingual texts, as will be reported in §6.1, we conducted five-times cross-validation separately for both data sets. We present the results in terms of the average accuracy  $A_{\mathbb{L}}$ , the ratio of the number of texts with a correctly identified language to  $|\mathcal{L}|$ .

### 5.2 Multilingual Texts (Test Data)

Multilingual texts were needed only to test the performance of the proposed method. In other words, we trained the model only through monolingual data, as mentioned above. This differs from the most similar previous study (Teahan, 2000), which required multilingual learning data.

The multilingual texts were generated artificially, since multilingual texts taken directly from the web have other issues besides segmentation. First, proper nouns in multilingual texts complicate the final judgment of language and segment borders. In practical application, therefore, texts for segmentation must be preprocessed by named entity recognition, which is beyond the scope of this work. Second, the sizes of text portions in multilingual web texts differ greatly, which would make it difficult to evaluate the overall performance of the proposed method in a uniform manner.

Consequently, we artificially generated two kinds of test sets from a monolingual corpus. The first is a set of multilingual texts, denoted as  $Test_1$ , such that each text is the conjunction of two portions in different languages. Here, the experiment is focused on segment border detection, which must segment the text into two parts, provided that there are two languages.  $Test_1$  includes test data for all language pairs, obtained by five-times cross-validation, giving  $25 \times |\mathcal{L}| \times (|\mathcal{L}| - 1)$  multilingual texts. Each portion of text for a single language consists of 100 characters taken from a random location within the test data.

The second kind of test set is a set of multilingual texts, denoted as  $Test_2$ , each consisting of  $k$  segments in different languages. For the experiment,  $k$  is not given to the procedure, and the task is to obtain  $k$  as well as  $\mathbb{B}$  and  $\mathbb{L}$  through recursion.  $Test_2$

<sup>8</sup><http://www.ohchr.org/EN/UDHR/Pages/Introduction.aspx>

<sup>9</sup><http://download.wikimedia.org/>

was generated through the following steps:

1. Choose  $k$  from among  $1, \dots, 5$ .
2. Choose  $k$  languages randomly from  $\mathcal{L}$ , where some of the  $k$  languages can overlap.
3. Perform five-times cross-validation on the texts of all languages. Choose a text length randomly from  $\{40, 80, 120, 160\}$ , and randomly select this many characters from the test data.
4. Shuffle the  $k$  languages and concatenate the text portions in the resultant order.

For this  $Test_2$  data set, every plot in the graphs shown in §6.2 was obtained by randomly averaging 1,000 tests.

By default, the possibility of segmentation is considered at every character offset in a text, which provides a *lower bound* for the proposed method. Although language change within the middle of a word does occur in real multilingual documents, it might seem more realistic to consider language change at word borders. Therefore, in addition to choosing  $\mathbb{B}$  from  $\{1, \dots, |X|\}$ , we also tested our approach under the constraint of choosing borders from *bordering locations*, which are the locations of spaces. In this case,  $\mathbb{B}$  is chosen from this subset of  $\{1, \dots, |X|\}$ , and, in step 3 above, text portions are generated so as to end at these bordering locations.

Given a multilingual text, we evaluate the outputs  $\mathbb{B}$  and  $\mathbb{L}$  through the following scores:

$P_{\mathbb{B}}/R_{\mathbb{B}}$ : Precision/recall of the borders detected (i.e., the correct borders detected, divided by the detected/correct border).

$P_{\mathbb{L}}/R_{\mathbb{L}}$ : Precision/recall of the languages detected (i.e., the correct languages detected, divided by the detected/correct language).

$P_s$  and  $R_s$  are obtained by changing the parameter  $\gamma$  given in formula (3), which ranges over  $1, 2, 4, \dots, 256$  bits. In addition, we verify the speed, i.e., the average time required for processing a text.

Although there are web pages consisting of texts in more than 2 languages, we rarely see a web page containing 5 languages at the same time. Therefore,  $Test_1$  reflects the most important case of 2 languages only, whereas  $Test_2$  reflects the case of multiple languages to demonstrate the general potential of the proposed approach.

The experiment reported here might seem like a case of over-specification, since all languages are considered equally likely to appear. Since our motivation has been to eliminate a portion in a major

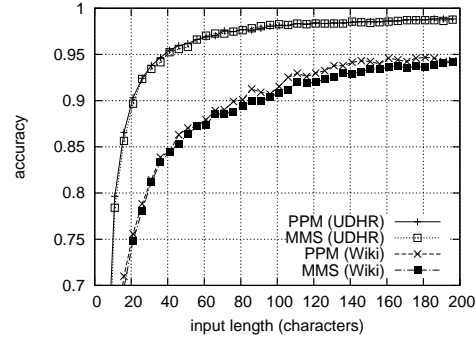


Figure 1: Accuracy of language identification for monolingual texts

language from the text, there could be a formulation specific to the problem. We consider it trivial, however, to specify such a narrow problem within our formulation, and it will lead to higher performance than that of the reported results, in any case. Therefore, we believe that our general formulation and experiment show the broadest potential of our approach to solving this problem.

## 6 Experimental Results

### 6.1 Language Identification Performance

We first show the performance of language identification using formula (2), which is used as the component of the text segmentation by language. Figure 1 shows the results for language identification of monolingual texts with the UDHR and Wiki test data. The horizontal axis indicates the size of the input text in characters, the vertical axis indicates the accuracy  $A_{\mathbb{L}}$ , and the graph contains four plots<sup>10</sup> for MMS and PPM for each set of data.

Overall, all plots rise quickly despite the severe conditions of a large number of languages (over 200), a small amount of input data, and a small amount of learning data. The results show that language identification through cross-entropy is promising.

Two further global tendencies can be seen. First, the performance was higher for UDHR than for Wiki. This is natural, since the content of Wikipedia is far broader than that of UDHR. In the case of UDHR, when the test data had a length of 40 characters, the accuracy was over 95% for both the PPM and the MMS methods. Second, PPM achieved

<sup>10</sup>The results for PPM and MMS for UDHR are almost the same, so the graph appears to contain only three plots.

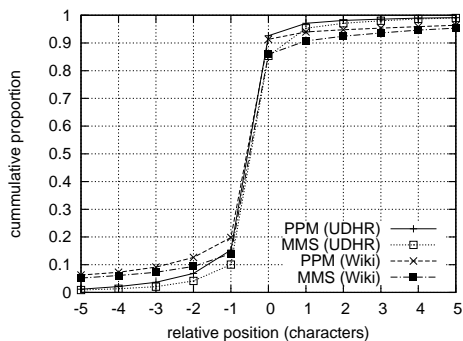


Figure 2: Cumulative distribution of segment borders

slightly better performance than did MMS. When the test data amounted to 100 characters, PPM achieved language identification with accuracy of about 91.4%. For MMS, the identification accuracy was a little less significant and was about 90.9% even with 100 characters of test data.

The amount of learning data seemed sufficient for both cases, with around 8,000 characters. In fact, we conducted tests with larger amounts of learning data and found a faster rise with respect to the input length, but the maximum possible accuracy did not show any significant increase.

Errors resulted from either noise or mistakes due to the language family. The Wikipedia test data was noisy, as mentioned in §5.1. As for language family errors, the test data includes many similar languages that are difficult even for humans to correctly judge. For example, Indonesian and Malay, Picard and Walloon, and Norwegian Bokmål and Nynorsk are all pairs representative of such confusion.

Overall, the language identification performance seems sufficient to justify its application to our main problem of text segmentation by language.

## 6.2 Text Segmentation by Language

First, we report the results obtained using the  $Test_1$  data set. Figure 2 shows the cumulative distribution obtained for segment border detection. The horizontal axis indicates the relative location by character with respect to the correct border at zero, and the vertical axis indicates the cumulative proportion of texts whose border is detected at that relative point. The figure shows four plots for all combinations of the two data sets and the two methods. Note that segment borders are judged by characters and *not* by bordering locations, as explained in §5.2.

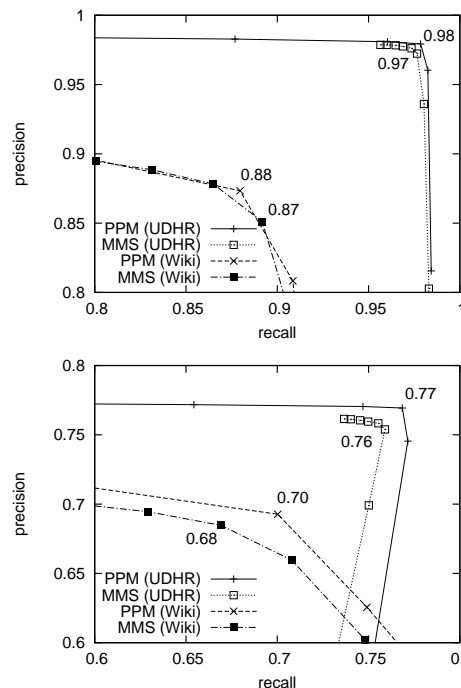


Figure 3:  $P_L/R_L$  (language, upper graph) and  $P_B/R_B$  (border, lower graph) results, where borders were taken from *any* character offset

Since the plots rise sharply at the middle of the horizontal axis, the borders were detected at or very near the correct place in many cases.

Next, we examine the results for  $Test_2$ . Figure 3 shows the two precision/recall graphs for language identification (upper graph) and segment border detection (lower graph), where borders were taken from any character offset. In each graph, the horizontal axis indicates precision and the vertical axis indicates recall. The numbers appearing in each figure are the maximum F-score values for each method and data set combination. As can be seen from these numbers, the language identification performance was high. Since the text portion size was chosen from among the values 40, 80, 120, or 160, the performance is comprehensible from the results shown in §6.1. Note also that PPM performed slightly better than did MMS.

For segment border performance (lower graph), however, the results were limited. The main reason for this is that both MMS and PPM tend to detect a border one character earlier than the correct location, as was seen in Figure 2. At the same time, much of the test data contains unrealistic borders

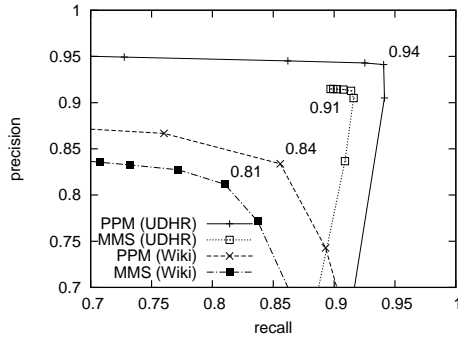


Figure 4:  $P_{\mathbb{B}}/R_{\mathbb{B}}$ , where borders were limited to spaces

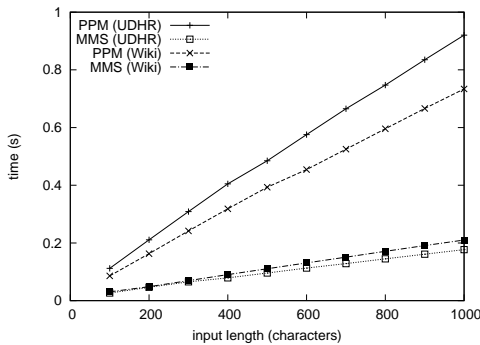


Figure 5: Average processing speed for a text

within a word, since the data was generated by concatenating two text portions with random borders. Therefore, we repeated the experiment with  $Test_2$  under the constraint that a segment border could occur only at a bordering location, as explained in §5.2. The results with this constraint were significantly better, as shown in Figure 4. The best result was for UDHR with PPM at 0.94<sup>11</sup>. We could also observe how PPM performed better at detecting borders in this case. In actual application, it would be possible to improve performance by relaxing the procedural conditions, such as by decreasing the number of language possibilities.

In this experiment for  $Test_2$ ,  $k$  ranged from 1 to 5, but the performance was not affected by the size of  $k$ . When the F-score was examined with respect to  $k$ , it remained almost equal to  $k$  in all cases. This shows how each recursion of formula (4) works almost independently, having segmentation and language identification functions that are both robust.

Lastly, we examine the speed of our method. Since  $|\mathcal{L}|$  is constant throughout the comparison,

<sup>11</sup>The language identification accuracy slightly increased as well, by 0.002.

the time should increase linearly with respect to the input length  $|X|$ , with increasing  $k$  having no effect. Figure 5 shows the speed for  $Test_2$  processing, with the horizontal axis indicating the input length and the vertical axis indicating the processing time. Here, all character offsets were taken into consideration, and the processing was done on a machine with a Xeon5650 2.66-GHz CPU. The results confirm that the complexity increased linearly with respect to the input length. When the text size became as large as several thousand characters, the processing time became as long as a second. This time could be significantly decreased by introducing constraints on the bordering locations and languages.

## 7 Conclusion

This article has presented a method for segmenting a multilingual text into segments, each in a different language. This task could serve for preprocessing of multilingual texts before applying language-specific analysis to each text. Moreover, the proposed method could be used to generate corpora in a variety of languages, since many texts in minor languages tend to contain chunks in a major language.

The segmentation task was modeled as an optimization problem of finding the best segment and language sequences to minimize the description length of a given text. An actual procedure for obtaining an optimal result through dynamic programming was proposed. Furthermore, we showed a way to decrease the computational complexity substantially, with each of our two methods having linear complexity in the input length.

Various empirical results were shown for language identification and segmentation. Overall, when segmenting a text with up to five random portions of different languages, where each portion consisted of 40 to 120 characters, the best F-scores for language identification and segmentation were 0.98 and 0.94, respectively.

For our future work, details of the methods must be worked out. In general, the proposed approach could be further applied to the actual needs of preprocessing and to generating corpora of minor languages.

## References

- Beatrice Alex, Amit Dubey, and Frank Keller. 2007. Using foreign inclusion detection to improve parsing performance. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 151–160.
- Beatrice Alex. 2005. An unsupervised system for identifying english inclusions in german text. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics, Student Research Workshop*, pages 133–138.
- T.C. Bell, J.G. Cleary, and I. H. Witten. 1990. *Text Compression*. Prentice Hall.
- Dario Benedetto, Emanuele Caglioti, and Vittorio Loreto. 2002. Language trees and zipping. *Physical Review Letters*, 88(4).
- Rudi Cilibrasi and Paul Vitányi. 2005. Clustering by compression. *IEEE Transactions on Information Theory*, 51(4):1523–1545.
- John G. Cleary and Ian H. Witten. 1984. Data compression using adaptive coding and partial string matching. *IEEE Transactions on Communications*, 32:396–402.
- Martin Farach, Michiel Noordewier, Serap Savari, Larry Shepp, Abraham J. Wyner, and Jacob Ziv. 1994. On the entropy of dna: Algorithms and measurements based on memory and rapid convergence. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 48–57.
- Gregory Grefenstette. 1995. Comparing two language identification schemes. In *Proceedings of 3rd International Conference on Statistical Analysis of Textual Data*, pages 263–268.
- Marti A. Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23(1):33–64.
- Patrick Juola. 1997. What can we do with small corpora? document categorization via cross-entropy. In *Proceedings of an Interdisciplinary Workshop on Similarity and Categorization*.
- Gen-itiro Kikui. 1996. Identifying the coding system and language of on-line documents on the internet. In *Proceedings of 16th International Conference on Computational Linguistics*, pages 652–657.
- Casanai Kruengkrai, Prapass Srichaivattana, Virach Sornlertlamvanich, and Hitoshi Isahara. 2005. Language identification based on string kernels. In *Proceedings of the 5th International Symposium on Communications and Information Technologies*, pages 926–929.
- Penelope Sibun and Jeffrey C. Reynar. 1996. Language identification: Examining the issues. In *Proceedings of 5th Symposium on Document Analysis and Information Retrieval*, pages 125–135.
- William J. Teahan and David J. Harper. 2001. Using compression-based language models for text categorization. In *Proceedings of the Workshop on Language Modeling and Information Retrieval*, pages 83–88.
- William John Teahan. 2000. Text classification and segmentation using minimum cross-entropy. In *RIAO*, pages 943–961.
- Jacob Ziv and Abraham Lempel. 1977. A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343.

## Appendix

This Appendix lists all the languages contained in our data sets, as summarized in Table 1.

## For UDHR

### Latin

Achinese, Achuar-Shiwiari, Adangme, Afrikaans, Aguaruna, Aja, Akuapem Akan, Akurio, Amahuaca, Amaraeri, Ambo-Pasco Quechua, Arabela, Arequipa-La Unión Quechua, Arpitan, Asante Akan, Asháninka, Ashéninka Pajonal, Asturian, Auvergnat Occitan, Ayacucho Quechua, Aymara, Baatonum, Balinese, Bambara, Baoulé, Basque, Bemba, Beti, Bikol, Bini, Bislama, Bokmål Norwegian, Bora, Bosnian, Breton, Buginese, Cajamarca Quechua, Calderón Highland Quichua, Candoshi-Shapra, Caquinte, Cashibo-Cacataibo, Cashinahua, Catalan, Cebuano, Central Kanuri, Central Mazahua, Central Nahuatl, Chamorro, Chamula Tzotzil, Chayahuita, Chickasaw, Chiga, Chokwe, Chuanqian Cluster Miao, Chuukese, Corsican, Cusco Quechua, Czech, Dagbani, Danish, Dendi, Ditammari, Dutch, Eastern Maninkakan, Emiliano-Romagnolo, English, Esperanto, Estonian, Ewe, Falam Chin, Fanti, Faroese, Fijian, Filipino, Finnish, Fon, French, Friulian, Ga, Gagauz, Galician, Ganda, Garifuna, Gen, German, Ghag Albanian, Gonja, Guarani, Güilá Zapotec, Haitian Creole, Haitian Creole (popular), Haka Chin, Hani, Hausa, Hawaiian, Hiligaynon, Huamalíes-Dos de Mayo Huánuco Quechua, Huautla Mazatec, Huaylas Ancash Quechua, Hungarian, Ibibio, Icelandic, Ido, Igbo, Iloko, Indonesian, Interlingua, Irish, Italian, Javanese, Jola-Fonyi, K'iche', Kabiyè, Kabuverdianu, Kalaallisut, Kaonde, Kaqchikel, Kasem, Kekchí, Kimbundu, Kinyarwanda, Kituba, Konzo, Kpelle, Krio, Kurdish, Lamnso', Languedocien Occitan, Latin, Latvian, Lingala, Lithuanian, Lozi, Luba-Lulua, Lunda, Luvale, Luxembourgish, Madurese, Makhuwa, Makonde, Malagasy, Maltese, Mam, Maori, Mapudungun, Margos-Yarowilca-Lauricocha Quechua, Marshallese, Mba, Mende, Metlatónoc Mixtec, Mezquital Otomi, Mi'kmaq, Miahuatlán Zapotec, Minangkabau, Mossi, Mozarabic, Murui Huitoto, Mískito, Ndonga, Nigerian Pidgin, Nomatsiguenga, North Junín Quechua, Northeastern Dinka, Northern Conchucos Ancash Quechua, Northern Qiangdong Miao, Northern Sami, Northern Kurdish, Nyamwezi, Nyanja, Nyemba, Nynorsk Norwegian, Nzima, Ojítlán Chinantec, Oromo, Palauan, Pampanga, Papanla Totonac, Pedi, Picard, Pichis Ashéninka, Pijin, Pipil, Pohnpeian, Polish, Portuguese, Pulaar, Purepecha, Páez, Quechua, Rarotongan, Romanian, Romansh, Romany, Rundi, Salinan, Samoan, San Luís Potosí Huastec, Sango, Sardinian, Scots, Scottish Gaelic, Serbian,

Serer, Seselwa Creole French, Sharanahua, Shipibo-Conibo, Shona, Slovak, Somali, Soninke, South Ndebele, Southern Dagaare, Southern Qiangdong Miao, Southern Sotho, Spanish, Standard Malay, Sukuma, Sundanese, Susu, Swahili, Swati, Swedish, Sāotomense, Tahitian, Tedim Chin, Tetum, Tidikelt Tamazight, Timne, Tiv, Toba, Tojolabal, Tok Pisin, Tonga (Tonga Islands), Tonga (Zambia), Tsonga, Tswana, Turkish, Tzeltal, Umbundu, Upper Sorbian, Urarina, Uzbek, Veracruz Huastec, Vili, Vlax Romani, Walloon, Waray, Wayuu, Welsh, Western Frisian, Wolof, Xhosa, Yagua, Yanasha', Yao, Yapese, Yoruba, Yucateco, Zhuang, Zulu

### **Cyrillic**

Abkhazian, Belarusian, Bosnian, Bulgarian, Kazakh, Macedonian, Ossetian, Russian, Serbian, Tuvinian, Ukrainian, Yakut

### **Arabic**

Standard Arabic

### **Other**

Japanese, Korean, Mandarin Chinese, Modern Greek

### **For Wiki**

#### **Latin**

Afrikaans, Albanian, Aragonese, Aromanian, Arpitan, Asturian, Aymara, Azerbaijani, Bambara, Banyumasan, Basque, Bavarian, Bislama, Bosnian, Breton, Català, Cebuano, Central Bikol, Chavacano, Cornish, Corsican, Crimean Tatar, Croatian, Czech, Danish, Dimli, Dutch, Dutch Low Saxon, Emiliano-Romagnolo, English, Esperanto, Estonian, Ewe, Extremaduran, Faroese, Fiji Hindi, Finnish, French, Friulian, Galician, German, Gilaki, Gothic, Guarani, Hai//om, Haitian, Hakka Chinese, Hawaiian, Hungarian, Icelandic, Ido, Igbo, Iloko, Indonesian, Interlingua, Interlingue, Irish, Italian, Javanese, Kabyle, Kalaallisut, Kara-Kalpak, Kashmiri, Kashubian, Kongo, Korean, Kurdish, Ladino, Latin, Latvian, Ligurian, Limburgan, Lingala, Lithuanian, Lojban, Lombard, Low German, Lower Sorbian, Luxembourgish, Malagasy, Malay, Maltese, Manx, Maori, Mazanderani, Min Dong Chinese, Min Nan Chinese, Nahuatl, Narom, Navajo, Neapolitan, Northern Sami, Norwegian, Norwegian Nynorsk, Novial, Occitan, Old English, Panganga, Pangasinan, Panjabi, Papiamentu, Pennsylvania German, Piemontese, Pitcairn-Norfolk, Polish, Portuguese, Pushto, Quechua, Romanian, Romansh, Samoan, Samogitian Lithuanian, Sardinian, Saterfriesisch, Scots, Scottish Gaelic, Serbo-Croatian, Sicilian, Silesian, Slovak, Slovenian, Somali, Spanish, Sranan Tongo, Sundanese, Swahili, Swati, Swedish, Tagalog, Tahitian, Tarantino Sicilian, Tatar, Tetum, Tok Pisin, Tonga (Tonga Islands), Tosk Albanian, Tsonga, Tswana, Turkish, Turkmen, Uighur, Upper Sorbian, Uzbek, Venda, Venetian, Vietnamese, Vlaams, Vlax Romani, Volapük, Võro, Walloon, Waray, Welsh, Western Frisian, Wolof, Yoruba, Zeeuws, Zulu

#### **Cyrillic**

Abkhazian, Bashkir, Belarusian, Bulgarian, Chuvash, Erzya, Kazakh, Kirghiz, Macedonian, Moksha, Moldovan, Mongolian, Old Belarusian, Ossetian, Russian, Serbian, Tajik, Udmurt, Ukrainian, Yakut

### **Arabic**

Arabic, Egyptian Arabic, Gilaki, Mazanderani, Persian, Pushto, Uighur, Urdu

### **Devanagari**

Bihari, Hindi, Marathi, Nepali, Newari, Sanskrit

### **Other**

Amharic, Armenian, Assamese, Bengali, Bishnupriya, Burmese, Central Khmer, Chinese, Classical Chinese, Dhivehi, Gan Chinese, Georgian, Gothic, Gujarati, Hebrew, Japanese, Kannada, Lao, Malayalam, Modern Greek, Official Aramaic, Panjabi, Sinhala, Tamil, Telugu, Thai, Tibetan, Wu Chinese, Yiddish, Yue Chinese

# Improve SMT Quality with Automatically Extracted Paraphrase Rules

Wei He<sup>1</sup>, Hua Wu<sup>2</sup>, Haifeng Wang<sup>2</sup>, Ting Liu<sup>1\*</sup>

<sup>1</sup>Research Center for Social Computing and Information  
Retrieval, Harbin Institute of Technology  
{whe, tliu}@ir.hit.edu.cn

<sup>2</sup>Baidu  
{wu\_hua, wanghaifeng}@baidu.com

## Abstract

We propose a novel approach to improve SMT via paraphrase rules which are automatically extracted from the bilingual training data. Without using extra paraphrase resources, we acquire the rules by comparing the source side of the parallel corpus with the target-to-source translations of the target side. Besides the word and phrase paraphrases, the acquired paraphrase rules mainly cover the structured paraphrases on the sentence level. These rules are employed to enrich the SMT inputs for translation quality improvement. The experimental results show that our proposed approach achieves significant improvements of 1.6~3.6 points of BLEU in the oral domain and 0.5~1 points in the news domain.

## 1 Introduction

The translation quality of the SMT system is highly related to the coverage of translation models. However, no matter how much data is used for training, it is still impossible to completely cover the unlimited input sentences. This problem is more serious for online SMT systems in real-world applications. Naturally, a solution to the coverage problem is to bridge the gaps between the input sentences and the translation models, either from the input side, which targets on rewriting the input sentences to the MT-favored expressions, or from

the side of translation models, which tries to enrich the translation models to cover more expressions.

In recent years, paraphrasing has been proven useful for improving SMT quality. The proposed methods can be classified into two categories according to the paraphrase targets: (1) enrich translation models to cover more bilingual expressions; (2) paraphrase the input sentences to reduce OOVs or generate multiple inputs. In the first category, He et al. (2011), Bond et al. (2008) and Nakov (2008) enriched the SMT models via paraphrasing the training corpora. Kuhn et al. (2010) and Max (2010) used paraphrases to smooth translation models. For the second category, previous studies mainly focus on finding translations for unknown terms using phrasal paraphrases. Callison-Burch et al. (2006) and Marton et al. (2009) paraphrase unknown terms in the input sentences using phrasal paraphrases extracted from bilingual and monolingual corpora. Mirkin et al. (2009) rewrite OOVs with entailments and paraphrases acquired from WordNet. Onishi et al. (2010) and Du et al. (2010) use phrasal paraphrases to build a word lattice to get multiple input candidates. In the above methods, only word or phrasal paraphrases are used for input sentence rewriting. No structured paraphrases on the sentence level have been investigated. However, the information in the sentence level is very important for disambiguation. For example, we can only substitute *play* with *drama* in a context related to *stage* or *theatre*. Phrasal paraphrase substitutions can hardly solve such kind of problems.

In this paper, we propose a method that rewrites

---

This work was done when the first author was visiting Baidu.

\*Correspondence author: tliu@ir.hit.edu.cn

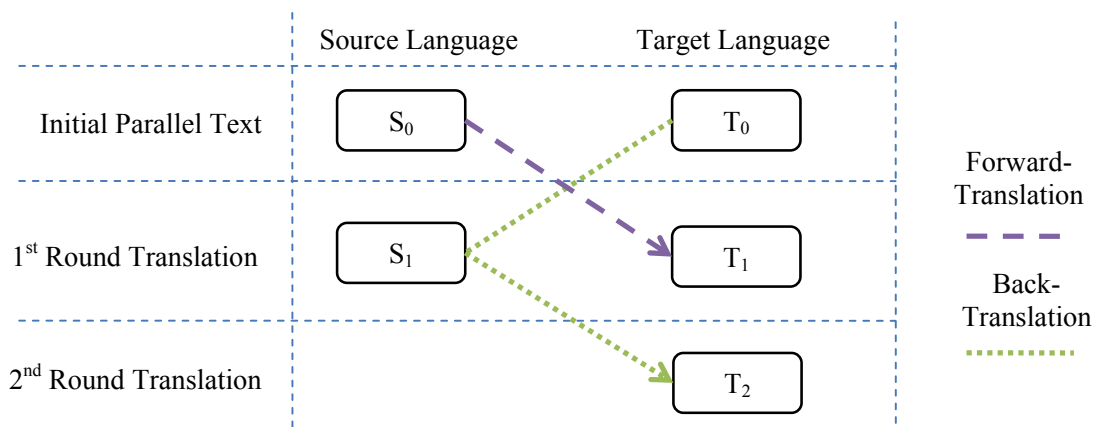


Figure 1: Procedure of Forward-Translation and Back-Translation.

the input sentences of the SMT system using automatically extracted paraphrase rules which can capture structures on sentence level in addition to paraphrases on the word or phrase level. Without extra paraphrase resources, a novel approach is proposed to acquire paraphrase rules from the bilingual training corpus based on the results of Forward-Translation and Back-Translation. The rules target on rewriting the input sentences to an MT-favored expression to ensure a better translation. The paraphrase rules cover all kinds of paraphrases on the word, phrase and sentence levels, enabling structure reordering, word or phrase insertion, deletion and substitution. The experimental results show that our proposed approach achieves significant improvements of 1.6~3.6 points of BLEU in the oral domain and 0.5~1 points in the news domain.

The remainder of the paper is organized as follows: Section 2 makes a comparison between the Forward-Translation and Back-Translation. Section 3 introduces our methods that extract paraphrase rules from the bilingual corpus of SMT. Section 4 describes the strategies for constructing word lattice with paraphrase rules. The experimental results and some discussions are presented in Section 5 and Section 6. Section 7 compares our work to the previous researches. Finally, Section 8 concludes the paper and suggests directions for future work.

## 2 Forward-Translation vs. Back-Translation

The Back-Translation method is mainly used for automatic MT evaluation (Rapp 2009). This

approach is very helpful when no target language reference is available. The only requirement is that the MT system needs to be bidirectional. The procedure includes translating a text into certain foreign language with the MT system (Forward-Translation), and translating it back into the original language with the same system (Back-Translation). Finally the translation quality of Back-Translation is evaluated by using the original source texts as references.

Sun et al. (2010) reported an interesting phenomenon: given a bilingual text, the Back-Translation results of the target sentences is better than the Forward-Translation results of the source sentences. Clearly, let  $(S_0, T_0)$  be the initial pair of bilingual text. A source-to-target translation system  $SYS\_ST$  and a target-to-source translation system  $SYS\_TS$  are trained using the bilingual corpus.  $FT(\cdot)$  is a Forward-Translation function, and  $BT(\cdot)$  is a function of Back-Translation which can be deduced with two rounds of translations:  $BT(s) = SYS\_TS(SYS\_ST(S))$ . In the first round of translation,  $S_0$  and  $T_0$  are fed into  $SYS\_ST$  and  $SYS\_TS$ , and we get  $T_1$  and  $S_1$  as translation results. In the second round, we translate  $S_1$  back into the target side with  $SYS\_ST$ , and get the translation  $T_2$ . The procedure is illustrated in Figure 1, which can also formally be described as:

1.  $T_1 = FT(S_0) = SYS\_ST(S_0)$ .
2.  $T_2 = BT(T_0)$ , which can be decomposed into two steps:  $S_1 = SYS\_TS(T_0)$ ,  $T_2 = SYS\_ST(S_1)$ .

Using  $T_0$  as reference, an interesting result is reported in Sun et al. (2010) that  $T_2$  achieves a higher score than  $T_1$  in automatic MT evaluation. This outcome is important because  $T_2$  is translated



No.	LHS	RHS
1	乘坐/ride X <sub>1</sub> 公共汽车/bus	乘坐/ride X <sub>1</sub> 巴士/bus
2	在/at X <sub>1</sub> 处/location 向左拐/turn left	向左拐/turn left 在/at X <sub>1</sub> 处/location
3	把/NULL X <sub>1</sub> 给/give 我/me	给/give 我/me X <sub>1</sub>
4	从/from X <sub>1</sub> 到/to X <sub>2</sub> 要/need 多长/how long 时间/time	要/need 花/spend 多长/how long 时间/time 从/from X <sub>1</sub> 到/to X <sub>2</sub>

Table 1: Examples of Chinese Paraphrase rules, together with English translations for every word

from a machine-generated text  $S_1$ , but  $T_1$  is translated from a human-write text  $S_0$ . Why the machine-generated text results in a better translation than the human-write text? Two possible reasons may explain this phenomenon: (1) in the first round of translation  $T_0 \rightarrow S_1$ , some target word orders are reserved due to the reordering failure, and these reserved orders lead to a better result in the second round of translation; (2) the text generated by an MT system is more likely to be matched by the reversed but homologous MT system.

Note that all the texts of  $S_0, S_1, S_2, T_0$  and  $T_1$  are sentence aligned because the initial parallel corpus  $(S_0, T_0)$  is aligned in the sentence level. The aligned sentence pairs in  $(S_0, S_1)$  can be considered as paraphrases. Since  $S_1$  has some MT-favored structures which may result in a better translation, an intuitive idea is whether we can learn these structures by comparing  $S_1$  with  $S_0$ . This is the main assumption of this paper. Taking  $(S_0, S_1)$  as paraphrase resource, we propose a method that automatically extracts paraphrase rules to capture the MT-favored structures.

### 3 Extraction of Paraphrase Rules

#### 3.1 Definition of Paraphrase Rules

We define a paraphrase rule as follows:

1. A paraphrase rule consists of two parts, left-hand-side (LHS) and right-hand-side (RHS). Both of LHS and RHS consist of non-terminals (slot) and terminals (words).
2. LHS must start/end with a terminal.
3. There must be at least one terminal between two non-terminals in LHS.

A paraphrase rule in the format of:

$$\text{LHS} \rightarrow \text{RHS}$$

which means the words matched by LHS can be paraphrased to RHS. Taking Chinese as a case

study, some examples of paraphrase rules are shown in Table 1.

#### 3.2 Selecting Paraphrase Sentence Pairs

Following the methods in Section 2, the initial bilingual corpus is  $(S_0, T_0)$ . We train a source-to-target PBMT system ( $SYS\_ST$ ) and a target-to-source PBMT system ( $SYS\_TS$ ) on the parallel corpus. Then a Forward-Translation is performed on  $S_0$  using  $SYS\_ST$ , and a Back-Translation is performed on  $T_0$  using  $SYS\_TS$  and  $SYS\_ST$ . As mentioned above, the detailed procedure is:  $T_1 = SYS\_ST(S_0)$ ,  $S_1 = SYS\_TS(T_0)$ ,  $T_2 = SYS\_ST(S_1)$ . Finally we compute BLEU (Papineni et al. 2002) score for every sentence in  $T_2$  and  $T_1$ , using the corresponding sentence in  $T_0$  as reference. If the sentence in  $T_2$  has a higher BLEU score than the aligned sentence in  $T_1$ , the corresponding sentences in  $S_0$  and  $S_1$  are selected as candidate paraphrase sentence pairs, which are used in the following steps of paraphrase extractions.

#### 3.3 Word Alignments Filtering

We can construct word alignment between  $S_0$  and  $S_1$  through  $T_0$ . On the initial corpus of  $(S_0, T_0)$ , we conduct word alignment with Giza++ (Och and Ney, 2000) in both directions and then apply the grow-diag-final heuristic (Koehn et al., 2005) for symmetrization. Because  $S_1$  is generated by feeding  $T_0$  into the PBMT system  $SYS\_TS$ , the word alignment between  $T_0$  and  $S_1$  can be acquired from the verbose information of the decoder.

The word alignments of  $S_0$  and  $S_1$  contain noises which are produced by either wrong alignment of GIZA++ or translation errors of  $SYS\_TS$ . To ensure the alignment quality, we use some heuristics to filter the alignment between  $S_0$  and  $S_1$ :

1. If two identical words are aligned in  $S_0$  and  $S_1$ , then remove all the other links to the two words.

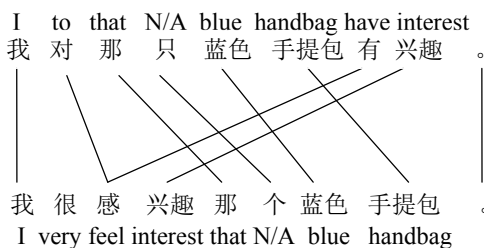
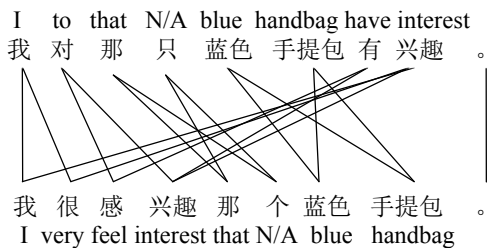


Figure 2: Example for Word Alignment Filtration

2. Stop words (including some function words and punctuations) can only be aligned to either stop words or null.

Figure 2 illustrates an example of using the heuristics to filter alignment.

### 3.4 Extracting Paraphrase Rules

From the word-aligned sentence pairs, we then extract a set of rules that are consistent with the word alignments. We use the rule extracting methods of Chiang (2005). Take the sentence pair in Figure 2 as an example, two initial phrase pairs  $PP_1 = \text{“那只蓝色手提包 ||| 那个蓝色手提包”}$  and  $PP_2 = \text{“对那只蓝色手提包有兴趣 ||| 我很感兴趣那个蓝色手提包”}$  are identified, and  $PP_1$  is contained by  $PP_2$ , then we could form the rule:

对  $X_1$  有兴趣  $\rightarrow$  很感兴趣  $X_1$   
to have interest very feel interest

## 4 Paraphrasing the Input Sentences

The extracted paraphrase rules aim to rewrite the input sentences to an MT-favored form which may lead to a better translation. However, it is risky to directly replace the input sentence with a paraphrased sentence, since the errors in automatic paraphrase substitution may jeopardize the translation result seriously. To avoid such damage, for a given input sentence, we first transform all paraphrase rules that match the input sentences to phrasal paraphrases, and then build a word lattice

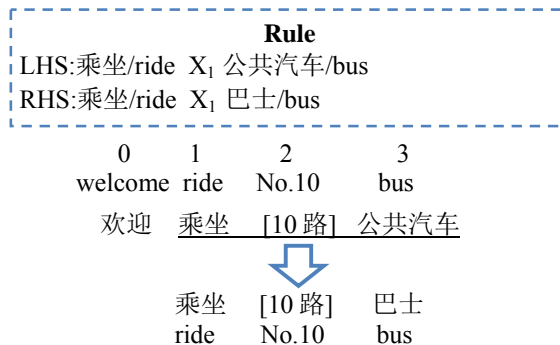


Figure 3: Example for Applying Paraphrase Rules for SMT decoder using the phrasal paraphrases. In this case, the decoder can search for the best result among all the possible paths.

The input sentences are first segmented into sub-sentences by punctuations. Then for each sub-sentence, the matched paraphrase rules are ranked according to: (1) the number of matched words; (2) the frequency of the paraphrase rule in the training data. Actually, the ranking strategy tends to select paraphrase rules that have more matched words (therefore less ambiguity) and higher frequency (therefore more reliable).

### 4.1 Applying Paraphrase Rules

Given an input sentence  $S$  and a paraphrase rule  $R \langle R_{LHS}, R_{RHS} \rangle$ , if  $S$  matches  $R_{LHS}$ , then the matched part can be replaced by  $R_{RHS}$ . An example for applying the paraphrase rules is illustrated in Figure 3.

From Figure 3, we can see that the words of position 1~3 are replaced to “乘坐 10 路 巴士”. Actually, only the words at position 3 and 4 are paraphrased to the word “巴士”, other words are left unchanged. Therefore, we can use a triple,  $\langle MIN\_RP\_TEXT, COVER\_START, COVER\_LEN \rangle$  ( $\langle \text{巴士}, 3, 1 \rangle$  in this example) to denote the paraphrase rule, which means the minimal text to replace is “巴士”, and the paraphrasing starts at position 3 and covers 1 words.

In this manner, all the paraphrase rules matched for a certain sentence can be converted to the format of  $\langle MIN\_RP\_TEXT, COVER\_START, COVER\_LEN \rangle$ , which can also be considered as phrasal paraphrases. Then the methods of building phrasal paraphrases into word lattice for SMT inputs can be used in our approaches.

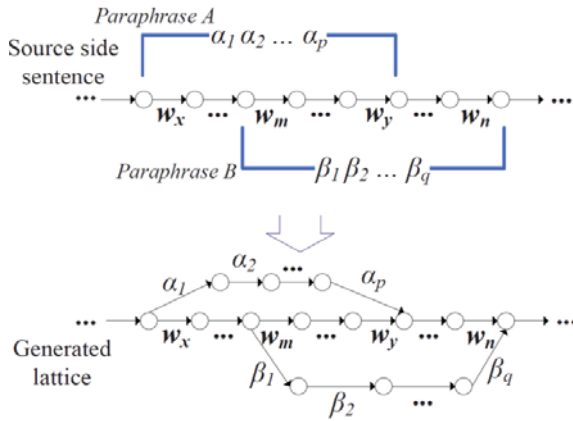


Figure 4: An example of lattice-based paraphrases for an input sentence.

#### 4.2 Construction of Paraphrase Lattice

Given an input sentence, all the matched paraphrase rules are converted to phrasal paraphrases first. Then we build the phrasal paraphrases into word lattices using the methods proposed by Du et al. (2010). The construction process takes advantage of the correspondence between detected phrasal paraphrases and positions of the original words in the input sentence, and then creates extra edges in the lattices to allow the decoder to consider paths involving the paraphrase words. An example is illustrated in Figure 4: given a sequence of words  $\{w_1, \dots, w_N\}$  as the input, two phrases  $\alpha = \{\alpha_1, \dots, \alpha_p\}$  and  $\beta = \{\beta_1, \dots, \beta_q\}$  are detected as paraphrases for  $P_1 = \{w_x, \dots, w_y\}$  ( $1 \leq x \leq y \leq N$ ) and  $P_2 = \{w_m, \dots, w_n\}$  ( $1 \leq m \leq n \leq N$ ) respectively. The following steps are taken to transform them into word lattices:

1. Transform the original source sentence into word lattice.  $N + 1$  nodes ( $\theta_k, 0 \leq k \leq N$ ) are created, and  $N$  edges labeled with  $w_i$  ( $1 \leq i \leq N$ ) are generated to connect them sequentially.
2. Generate extra nodes and edges for each of the paraphrases. Taking  $\alpha$  as an example, firstly,  $p - 1$  nodes are created, and then  $p$  edges labeled with  $\alpha_j$  ( $1 \leq j \leq p$ ) are generated to connect node  $\theta_{x-1}$ ,  $p-1$  nodes and  $\theta_{y-1}$ .

Via step 2, word lattices are generated by adding new nodes and edges coming from paraphrases.

#### 4.3 Weight Estimation

The weights of new edges in the lattices are estimated by an empirical method base on ranking positions. Following Du et al. (2010), supposing that  $E = \{e_1, \dots, e_k\}$  are a set of new edges constructed from  $k$  paraphrase rules, which are sorted in a descending order. Then the weight for an edge  $e_i$  is calculated as:

$$w(e_i) = \frac{1}{k + i} \quad (1 \leq i \leq k)$$

where  $k$  is a predefined tradeoff parameter between decoding speed and the number of potential paraphrases being considered.

### 5 Experiments

#### 5.1 Experimental Data

In our experiments, we used Moses (Koehn et al., 2007) as the baseline system which can support lattice decoding. The alignment was obtained using GIZA++ (Och and Ney, 2003) and then we symmetrized the word alignment using the grow-diag-final heuristic. Parameters were tuned using Minimum Error Rate Training (Och, 2003). To comprehensively evaluate the proposed methods in different domains, two groups of experiments were carried out, namely, the oral group ( $G_{\text{oral}}$ ) and the news group ( $G_{\text{news}}$ ). The experiments were conducted in both Chinese-English and English-Chinese directions for the oral group, and Chinese-English direction for the news group. The English sentences were all tokenized and lowercased, and the Chinese sentences were segmented into words by Language Technology Platform (LTP)<sup>1</sup>. We used SRILM<sup>2</sup> for the training of language models (5-gram in all the experiments). The metrics for automatic evaluation were BLEU<sup>3</sup> and TER<sup>4</sup> (Snover et al., 2005).

The detailed statistics of the training data in  $G_{\text{oral}}$  are showed in Table 2. For the bilingual corpus, we used the BTEC and PIVOT data of IWSLT 2008, HIT corpus<sup>5</sup> and other Chinese LDC (CLDC)

<sup>1</sup> <http://ir.hit.edu.cn/ltp/>

<sup>2</sup> <http://www.speech.sri.com/projects/srilm/>

<sup>3</sup> <http://jaguar.ncsl.nist.gov/mt/resources/mteval-v13a.pl>

<sup>4</sup> <http://www.umiacs.umd.edu/~snover/terp/>

<sup>5</sup> The HIT corpus contains the CLDC Olympic corpus (2004-863-008) and the other HIT corpora available at <http://mitlab.hit.edu.cn/index.php/resources/29-the-resource/111-share-bilingual-corpus.html>.

Corpus	#Sen. pairs	#Ch. words	#En words
BETC	19,972	174k	190k
PIVOT	20,000	162k	196k
HIT	80,868	788k	850k
CLDC	190,447	1,167k	1,898k
Tanaka	149,207	-	1,375k

Table 2: Statistics of training data in  $G_{\text{oral}}$

	Corpus	#Sen.	#Ref.
develop	CSTAR03 test set	506	16
	IWSLT06 dev set	489	7
test	IWSLT04 test set	500	16
	IWSLT05 test set	506	16
	IWSLT06 test set	500	7
	IWSLT07 test set	489	6

Table 3: Statistics of test/develop sets in  $G_{\text{oral}}$

	Corpus	#Sen.	#Ref.
develop	NIST 2002	878	10
	NIST 2005	1,082	4
test	NIST 2004	1,788	5
	NIST 2006	1,664	4
	NIST 2008	1,357	4

Table 4: Statistics of test/develop sets in  $G_{\text{news}}$

corpora, including the Chinese-English Sentence Aligned Bilingual Corpus (CLDC-LAC-2003-004) and the Chinese-English Parallel Corpora (CLDC-LAC-2003-006). We trained a Chinese language model for the E-C translation on the Chinese part of the bi-text. For the English language model of C-E translation, an extra corpus named Tanaka was used besides the English part of the bilingual corpora. For testing and developing, we used six Chinese-English development corpora of IWSLT 2008. The statistics are shown in Table 3.

In detail, we chose CSTAR03-test and IWSLT06-dev as the development set; and used IWSLT04-test, IWSLT05-test, IWSLT06-dev and IWSLT07-test for testing. For English-Chinese evaluation, we used IWSLT English-Chinese MT evaluation 2005 as the test set. Due to the lacking of development set, we did not tune parameters on English-Chinese side, instead, we just used the default parameters of Moses.

In the experiments of the news group, we used the Sinorama and FBIS corpora (LDC2005T10 and LDC2003E14) for bilingual corpus. After tokenization and filtering, this bilingual corpus contained 319,694 sentence pairs (7.9M tokens on

Chinese side and 9.2M tokens on English side). We trained a 5-gram language model on the English side of the bi-text. The system was tested using the Chinese-English MT evaluation sets of NIST 2004, NIST 2006 and NIST 2008. For development, we used the Chinese-English MT evaluation sets of NIST 2002 and NIST 2005. Table 4 shows the statistics of test/development sets used in the news group.

## 5.2 Results

We extract both Chinese and English rules in  $G_{\text{oral}}$ , and Chinese paraphrase rules in  $G_{\text{news}}$  by comparing the results of Forward-Translation and Back-Translation as described in Section 3. During the extraction, some heuristics are used to ensure the quality of paraphrase rules. Take the extraction of Chinese paraphrase rules in  $G_{\text{oral}}$  as a case study. Suppose  $(C_0, E_0)$  are the initial bilingual corpus of  $G_{\text{oral}}$ . A Chinese-English and an English-Chinese MT system are trained on  $(C_0, E_0)$ . We perform Back-Translation on  $E_0$  ( $E_0 \xrightarrow{E \text{ to } C} C_1 \xrightarrow{C \text{ to } E} E_2$ ), and Forward-Translation on  $C_0$  ( $C_0 \xrightarrow{C \text{ to } E} E_1$ ). Suppose  $e_{1i}$  and  $e_{2i}$  are two aligned sentences in  $E_1$  and  $E_2$ ,  $c_{0i}$  and  $c_{1i}$  are the corresponding sentences in  $C_0$  and  $C_1$ .  $(c_{0i}, c_{1i})$  are selected for the extraction of paraphrase rules if two conditions are satisfied: (1)  $\text{BLEU}(e_{2i}) - \text{BLEU}(e_{1i}) > \theta_1$ , and (2)  $\text{BLEU}(e_{2i}) > \theta_2$ , where  $\text{BLEU}(\cdot)$  is a function for computing BLEU score;  $\theta_1$  and  $\theta_2$  are thresholds for balancing the rules number and the quality of paraphrase rules. In our experiment,  $\theta_1$  and  $\theta_2$  are empirically set to 0.1 and 0.3.

As a result, we extract 912,625 Chinese and 1,116,375 English paraphrase rules for  $G_{\text{oral}}$ , and for  $G_{\text{news}}$  the number of Chinese paraphrase rules is 2,877,960. Then we use the extracted paraphrase rules to improve SMT by building word lattices for the input sentences.

The Chinese-English experimental results of  $G_{\text{oral}}$  and  $G_{\text{news}}$  are shown in Table 5 and Table 6, respectively. It can be seen that our method outperforms the baselines in both oral and news domains. Our system gains significant improvements of 1.6~3.6 points of BLEU in the oral domain, and 0.5~1 points of BLEU in the news domain. Figure 5 shows the effect of considered paraphrases (k) in the step of building

Model	BLEU				TER			
	iwslt 04	iwslt 05	iwslt 06	iwslt 07	iwslt 04	iwslt 05	iwslt 06	iwslt 07
baseline	0.5353	0.5887	0.2765	0.3977	0.3279	0.2874	0.5559	0.4390
para. improved	<b>0.5712</b>	<b>0.6107</b>	<b>0.2924</b>	<b>0.4193</b>	<b>0.3055</b>	<b>0.2722</b>	<b>0.5374</b>	<b>0.4217</b>

Table 5: Experimental results of  $G_{\text{oral}}$  in Chinese-English direction

Model	BLEU			TER		
	nist 04	nist 06	nist 08	nist 04	nist 06	nist 08
baseline	0.2795	0.2389	0.1933	0.6554	0.6515	0.6652
para. improved	<b>0.2891</b>	<b>0.2485</b>	<b>0.1978</b>	<b>0.6451</b>	<b>0.6407</b>	<b>0.6582</b>

Table 6: Experimental results of  $G_{\text{news}}$  in Chinese-English direction

model	IWSLT 2005	
	BLEU	TER
baseline	0.4644	0.4164
para. improved	<b>0.4853</b>	<b>0.3883</b>

Table 7: Experimental results of  $G_{\text{oral}}$  in English-Chinese direction

trans. para.	improve	comparable	worsen	total
correct	36	20	4	60
incorrect	1	9	14	24

Table 8: Human analysis of the paraphrasing results in IWSLT 2007 CE translation

word lattices. The result of English-Chinese experiments in  $G_{\text{oral}}$  is shown in Table 7.

## 6 Discussion

We make a detailed analysis on the Chinese-English translation results that are affected by our paraphrase rules. The aim is to investigate what kinds of paraphrases have been captured in the rules. Firstly the input path is recovered from the translation results according to the tracing information of the decoder, and therefore we can examine which path is selected by the SMT decoder from the paraphrase lattice. A human annotator is asked to judge whether the recovered paraphrase sentence keeps the same meaning as the original input. Then the annotator compares the baseline translation with the translations proposed by our approach. The analysis is carried out on the IWSLT 2007 Chinese-English test set, 84 out of 489 input sentences have been affected by paraphrases, and the statistic of human evaluation is shown in Table 8.

It can be seen in Table 8 that the paraphrases achieve a relatively high accuracy, 60 (71.4%)

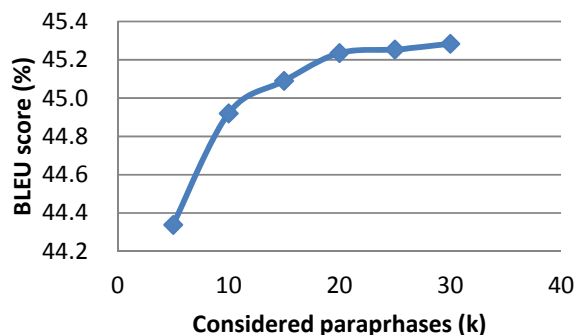


Figure 5: Effect of considered paraphrases (k) on BLEU score

paraphrased sentences retain the same meaning, and the other 24 (28.6%) are incorrect. Among the 60 correct paraphrases, 36 sentences finally result in an improved translation. We further analyze these paraphrases and the translation results to investigate what kinds of transformation finally lead to the translation quality improvement. The paraphrase variations can be classified into four categories:

- (1) Reordering: The original source sentences are reordered to be similar to the order of the target language.
- (2) Word substitution: A phrase with multi-word translations is replaced by a phrase with a single-word translation.
- (3) Recovering omitted words: Ellipsis occurs frequently in spoken language. Recovering the omitted words often leads to a better translation.
- (4) Removing redundant words: Mostly, translating redundant words may confuse the SMT system and would be unnecessary. Removing redundant words can mitigate this problem.

Cate.	Num	Original Sentence/Translation	Paraphrased Sentence/Translation
(1)	11	香烟/cigarette 可以/can 免税/duty-free 带/take 多少/how much 支/N/A ? what a cigarette can i take duty-free ?	多少/how much 香烟/cigarettes 可以/can 免税/duty-free 带/take 支/N/A ? how many cigarettes can i take duty-free one ?
(2)	18	你/you 有/have 多久/how long 的/N/A 教学/teaching 经验/experience ? you have how long teaching experience ?	你/you 有/have 多少/how much 教学/teaching 经验/experience ? how much teaching experience you have ?
(3)	10	需要/need 押金/deposit 吗/N/A ? you need a deposit ?	你/you 需要/need 押金/deposit 吗/N/A ? do you need a deposit ?
(4)	4	戒指/ring 掉/fall 进/into 洗脸池/washbasin 里/in 了/N/A 。 ring off into the washbasin is in .	戒指/ring 掉/fall 进/into 洗脸池/washbasin 了/N/A 。 ring off into the washbasin .

Table 9: Examples for classification of paraphrase rules

Four examples for category (1), (2), (3) and (4) are shown in Table 9, respectively. The numbers in the second column indicates the number of the sentences affected by the rules, among the 36 sentences with improved paraphrasing and translation. A sentence can be classified into multiple categories. Except category (2), the other three categories cannot be detected by the previous approaches, which verify our statement that our rules can capture structured paraphrases on the sentence level in addition to the paraphrases on the word or phrase level.

Not all the paraphrased results are correct. Sometimes an ill paraphrased sentence can produce better translations. Take the first line of Table 9 as an example, the paraphrased sentence “多少/How many 香烟/cigarettes 可以/can 免税/duty-free 带/take 支/NULL” is not a fluent Chinese sentence, however, the rearranged word order is closer to English, which finally results in a much better translation.

## 7 Related Work

Previous studies on improving SMT using paraphrase rules focus on hand-crafted rules. Nakov (2008) employs six rules for paraphrasing the training corpus. Bond et al. (2008) use grammars to paraphrase the source side of training data, covering aspects like word order and minor lexical variations (tenses etc.) but not content words. The paraphrases are added to the source side of the corpus and the corresponding target sentences are duplicated.

A disadvantage for hand-crafted paraphrase rules is that it is language dependent. In contrast, our method that automatically extracted paraphrase

rules from bilingual corpus is flexible and suitable for any language pairs.

Our work is similar to Sun et al. (2010). Both tried to capture the MT-favored structures from bilingual corpus. However, a clear difference is that Sun et al. (2010) captures the structures implicitly by training an MT system on ( $S_0$ ,  $S_1$ ) and “translates” the SMT input to an MT-favored expression. Actually, the rewriting process is considered as a black box in Sun et al. (2010). In this paper, the MT-favored expressions are captured explicitly by automatically extracted paraphrase rules. The advantages of the paraphrase rules are: (1) Our method can explicitly capture the structure information in the sentence level, enabling global reordering, which is impossible in Sun et al. (2010). (2) For each rule, we can control its quality automatically or manually.

## 8 Conclusion

In this paper, we propose a novel method for extracting paraphrase rules by comparing the source side of bilingual corpus to the target-to-source translation of the target side. The acquired paraphrase rules are employed to enrich the SMT inputs, which target on rewriting the input sentences to an MT-favored form. The paraphrase rules cover all kinds of paraphrases on the word, phrase and sentence levels, enabling structure reordering, word or phrase insertion, deletion and substitution. Experimental results show that the paraphrase rules can improve SMT quality in both the oral and news domains. The manual investigation on oral translation results indicate that the paraphrase rules capture four kinds of MT-favored transformation to ensure translation quality improvement.

## Acknowledgement

This work was supported by National Natural Science Foundation of China (NSFC) (61073126, 61133012), 863 High Technology Program (2011AA01A207).

## References

- Francis Bond, Eric Nichols, Darren Scott Appling, and Michael Paul. 2008. Improving Statistical Machine Translation by Paraphrasing the Training Data. In Proceedings of the IWSLT, pages 150–157.
- Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved Statistical Machine Translation Using Paraphrases. In Proceedings of NAACL, pages 17-24.
- David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In Proceedings of ACL, pages 263–270.
- Jinhua Du, Jie Jiang, Andy Way. 2010. Facilitating Translation Using Source Language Paraphrase Lattices. In Proceedings of EMNLP, pages 420-429.
- Wei He, Shiqi Zhao, Haifeng Wang and Ting Liu. 2011. Enriching SMT Training Data via Paraphrasing. In Proceedings of IJCNLP, pages 803-810.
- Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In Proceedings of HLT/NAACL, pages 48–54
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In Proceedings of EMNLP, pages 388-395.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne, and David Talbot. 2005. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In Proceedings of IWSLT.
- 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 the ACL Demo and Poster Sessions, pages 177–180.
- Roland Kuhn, Boxing Chen, George Foster and Evan Stratford. 2010. Phrase Clustering for Smoothing TM Probabilities-or, How to Extract Paraphrases from Phrase Tables. In Proceedings of COLING, pages 608–616.
- Yuval Marton, Chris Callison-Burch, and Philip Resnik. 2009. Improved Statistical Machine Translation Using Monolingually-Derived Paraphrases. In Proceedings of EMNLP, pages 381-390.
- Aurélien Max. 2010. Example-Based Paraphrasing for Improved Phrase-Based Statistical Machine Translation. In Proceedings of EMNLP, pages 656-666.
- Shachar Mirkin, Lucia Specia, Nicola Cancedda, Ido Dagan, Marc Dymetman, Idan Szpektor. 2009. Source-Language Entailment Modeling for Translation Unknown Terms. In Proceedings of ACL, pages 791-799.
- Preslav Nakov. 2008. Improved Statistical Machine Translation Using Monolingual Paraphrases. In Proceedings of ECAI, pages 338-342.
- Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In Proceedings of ACL, pages 440-447.
- Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In Proceedings of ACL, pages 160-167.
- Takashi Onishi, Masao Utiyama, Eiichiro Sumita. 2010. Paraphrase Lattice for Statistical Machine Translation. In Proceedings of ACL, pages 1-5.
- Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In Proceedings of ACL, pages 311-318.
- Reinhard Rapp. 2009. The Back-translation Score: Automatic MT Evaluation at the Sentence Level without Reference Translations. In Proceedings of ACL-IJCNLP, pages 133-136.
- Matthew Snover, Bonnie J. Dorr, Richard Schwartz, John Makhoul, Linnea Micciulla, and Ralph Weischedel. 2005. A study of translation error rate with targeted human annotation. Technical Report LAMP-TR-126, CS-TR-4755, UMIACS-TR-2005-58, University of Maryland, July, 2005.
- Yanli Sun, Sharon O'Brien, Minako O'Hagan and Fred Hollowood. 2010. A Novel Statistical Pre-Processing Model for Rule-Based Machine Translation System. In Proceedings of EAMT, 8pp.

# Ecological Evaluation of Persuasive Messages Using Google AdWords

**Marco Guerini**

Trento-Rise

Via Sommarive 18, Povo

Trento — Italy

marco.guerini@trentorise.eu

**Carlo Strapparava**

FBK-Irst

Via Sommarive 18, Povo

Trento — Italy

strappa@fbk.eu

**Oliviero Stock**

FBK-Irst

Via Sommarive 18, Povo

Trento — Italy

stock@fbk.eu

## Abstract

In recent years there has been a growing interest in crowdsourcing methodologies to be used in experimental research for NLP tasks. In particular, evaluation of systems and theories about persuasion is difficult to accommodate within existing frameworks. In this paper we present a new cheap and fast methodology that allows fast experiment building and evaluation with fully-automated analysis at a low cost. The central idea is exploiting existing commercial tools for advertising on the web, such as Google AdWords, to measure message impact in an ecological setting. The paper includes a description of the approach, tips for how to use AdWords for scientific research, and results of pilot experiments on the impact of affective text variations which confirm the effectiveness of the approach.

## 1 Introduction

In recent years there has been a growing interest in finding new cheap and fast methodologies to be used in experimental research, for, but not limited to, NLP tasks. In particular, approaches to NLP that rely on the use of web tools - for crowdsourcing long and tedious tasks - have emerged. Amazon Mechanical Turk, for example, has been used for collecting annotated data (Snow et al., 2008). However approaches *a la* Mechanical Turk might not be suitable for all tasks.

In this paper we focus on evaluating systems and theories about persuasion, see for example (Fogg, 2009) or the survey on persuasive NL generation studies in (Guerini et al., 2011a). Measuring the

impact of a message is of paramount importance in this context, for example how affective text variations can alter the persuasive impact of a message.

The problem is that evaluation experiments represent a bottleneck: they are expensive and time consuming, and recruiting a high number of human participants is usually very difficult.

To overcome this bottleneck, we present a specific cheap and fast methodology to automatize large-scale evaluation campaigns. This methodology allows us to crowdsource experiments with thousands of subjects for a few euros in a few hours, by tweaking and using existing commercial tools for advertising on the web. In particular we make reference to the AdWords Campaign Experiment (ACE) tool provided within the Google AdWords suite. One important aspect of this tool is that it allows for real-time fully-automated data analysis to discover statistically significant phenomena. It is worth noting that this work originated in the need to evaluate the impact of short persuasive messages, so as to assess the effectiveness of different linguistic choices. Still, we believe that there is further potential for opening an interesting avenue for experimentally exploring other aspects of the wide field of pragmatics.

The paper is structured as follows: Section 2 discusses the main advantages of ecological approaches using Google ACE over traditional lab settings and state-of-the-art crowdsourcing methodologies. Section 3 presents the main AdWords features. Section 4 describes how AdWords features can be used for defining message persuasiveness metrics and what kind of stimulus characteristics can be evaluated. Finally Sections 5 and 6 describe how to build up an



experimental scenario and some pilot studies to test the feasibility of our approach.

## 2 Advantages of Ecological Approaches

Evaluation of the effectiveness of persuasive systems is very expensive and time consuming, as the STOP experience showed (Reiter et al., 2003): designing the experiment, recruiting subjects, making them take part in the experiment, dispensing questionnaires, gathering and analyzing data.

Existing methodologies for evaluating persuasion are usually split in two main sets, depending on the setup and domain: (i) long-term, in the field evaluation of behavioral change (as the STOP example mentioned before), and (ii) lab settings for evaluating short-term effects, as in (Andrews et al., 2008). While in the first approach it is difficult to take into account the role of external events that can occur over long time spans, in the second there are still problems of recruiting subjects and of time consuming activities such as questionnaire gathering and processing.

In addition, sometimes carefully designed experiments can fail because: (i) effects are too subtle to be measured with a limited number of subjects or (ii) participants are not engaged enough by the task to provoke usable reactions, see for example what reported in (Van Der Sluis and Mellish, 2010). Especially the second point is awkward: in fact, subjects can actually be convinced by the message to which they are exposed, but if they feel they do not care, they may not “react” at all, which is the case in many artificial settings. To sum up, the main problems are:

1. Time consuming activities
2. Subject recruitment
3. Subject motivation
4. Subtle effects measurements

### 2.1 Partial Solution - Mechanical Turk

A recent trend for behavioral studies that is emerging is the use of Mechanical Turk (Mason and Suri, 2010) or similar tools to overcome part of these limitations - such as subject recruitment. Still we believe that this poses other problems in assessing behavioral changes, and, more generally, persuasion effects. In fact:

1. Studies must be as ecological as possible, i.e. conducted in real, even if controlled, scenarios.
2. Subjects should be neither aware of being observed, nor biased by external rewards.

In the case of Mechanical Turk for example, subjects are willingly undergoing a process of being tested on their skills (e.g. by performing annotation tasks). Cover stories can be used to soften this awareness effect, nonetheless the fact that subjects are being paid for performing the task renders the approach unfeasible for behavioral change studies. It is necessary that the only reason for behavior induction taking place during the experiment (filling a form, responding to a questionnaire, clicking on an item, etc.) is the exposition to the experimental stimuli, not the external reward. Moreover, Mechanical Turk is based on the notion of a “gold standard” to assess contributors reliability, but for studies concerned with persuasion it is almost impossible to define such a reference: there is no “right” action the contributor can perform, so there is no way to assess whether the subject is performing the action because induced to do so by the persuasive strategy, or just in order to receive money. On the aspect of how to handle subject reliability in coding tasks, see for example the method proposed in (Negri et al., 2010).

### 2.2 Proposed Solution - Targeted Ads on the Web

Ecological studies (e.g. using Google AdWords) offer a possible solution to the following problems:

1. Time consuming activities: apart from experimental design and setup, all the rest is automatically performed by the system. Experiments can yield results in a few hours as compared to several days/weeks.
2. Subject recruitment: the potential pool of subjects is the entire population of the web.
3. Subject motivation: ads can be targeted exactly to those persons that are, in that precise moment throughout the world, most interested in the topic of the experiment, and so potentially more prone to react.
4. Subject unaware, unbiased: subjects are totally unaware of being tested, testing is performed during their “natural” activity on the web.

5. Subtle effects measurements: if there are not enough subjects, just wait for more ads to be displayed, or focus on a subset of even more interested people.

Note that similar ecological approaches are beginning to be investigated: for example in (Aral and Walker, 2010) an approach to assessing the social effects of content features on an on-line community is presented. A previous approach that uses AdWords was presented in (Guerini et al., 2010), but it crowd-sourced only the running of the experiment, not data manipulation and analysis, and was not totally controlled for subject randomness.

### 3 AdWords Features

Google AdWords is Google's advertising program. The central idea is to let advertisers display their messages only to relevant audiences. This is done by means of keyword-based contextualization on the Google network, divided into:

- Search network: includes Google search pages, search sites and properties that display search results pages (SERPs), such as Froogle and Earthlink.
- Display network: includes news pages, topic-specific websites, blogs and other properties - such as Google Mail and The New York Times.

When a user enters a query like "cruise" in the Google search network, Google displays a variety of relevant pages, along with ads that link to cruise trip businesses. To be displayed, these ads must be associated with relevant keywords selected by the advertiser.

Every advertiser has an AdWords account that is structured like a pyramid: (i) account, (ii) campaign and (iii) ad group. In this paper we focus on ad groups. Each grouping gathers similar keywords together - for instance by a common theme - around an ad group. For each ad group, the advertiser sets a cost-per-click (CPC) bid. The CPC bid refers to the amount the advertiser is willing to pay for a click on his ad; the cost of the actual click instead is based on its quality score (a complex measure out of the scope of the present paper).

For every ad group there could be multiple ads to be served, and there are many AdWords measure-

ments for identifying the performance of each single ad (its persuasiveness, from our point of view):

- CTR, Click Through Rate: measures the number of clicks divided by the number of impressions (i.e. the number of times an ad has been displayed in the Google Network).
- Conversion Rate: if someone clicks on an ad, and buys something on your site, that click is a conversion from a site visit to a sale. Conversion rate equals the number of conversions divided by the number of ad clicks.
- ROI: Other conversions can be page views or signups. By assigning a value to a conversion the resulting conversions represents a return on investment, or ROI.
- Google Analytics Tool: Google Analytics is a web analytics tool that gives insights into website traffic, like number of visited pages, time spent on the site, location of visitors, etc.

So far, we have been talking about text ads, - Google's most traditional and popular ad format - because they are the most useful for NLP analysis. In addition there is also the possibility of creating the following types of ads:

- Image (and animated) ads
- Video ads
- Local business ads
- Mobile ads

The above formats allow for a greater potential to investigate persuasive impact of messages (other than text-based) but their use is beyond the scope of the present paper<sup>1</sup>.

### 4 The ACE Tool

AdWords can be used to design and develop various metrics for fast and *fully-automated* evaluation experiments, in particular using the ACE tool.

This tool - released in late 2010 - allows testing, from a marketing perspective, if any change made to a promotion campaign (e.g. a keyword bid) had a statistically measurable impact on the campaign itself. Our primary aim is slightly different: we are

---

<sup>1</sup>For a thorough description of the AdWords tool see: <https://support.google.com/adwords/>

interested in testing how different messages impact (possibly different) audiences. Still the ACE tool goes exactly in the direction we aim at, since it incorporates *statistically significant testing* and allows avoiding many of the tweaking and tuning actions which were necessary before its release.

The ACE tool also introduces an option that was not possible before, that of *real-time testing* of statistical significance. This means that it is no longer necessary to define a-priori the sample size for the experiment: as soon as a meaningful statistically significant difference emerges, the experiment can be stopped.

Another advantage is that the statistical knowledge to evaluate the experiment is no longer necessary: the researcher can focus only on setting up proper experimental designs<sup>2</sup>.

The limit of the ACE tool is that it only allows A/B testing (single split with one control and one experimental condition) so for experiments with more than two conditions or for particular experimental settings that do not fit with ACE testing boundaries (e.g. cross campaign comparisons) we suggest taking (Guerini et al., 2010) as a reference model, even if the experimental setting is less controlled (e.g. subject randomness is not equally guaranteed as with ACE).

Finally it should be noted that even if ACE allows only A/B testing, it permits the decomposition of almost any variable affecting a campaign experiment in its basic dimensions, and then to segment such dimensions according to control and experimental conditions. As an example of this powerful option, consider Tables 3 and 6 where control and experimental conditions are compared against every single keyword and every search network/ad position used for the experiments.

## 5 Evaluation and Targeting with ACE

Let us consider the design of an experiment with 2 conditions. First we create an ad Group with 2 competing messages (one message for each condition). Then we choose the serving method (in our opinion the *rotate* option is better than *optimize*, since it

---

<sup>2</sup>Additional details about ACE features and statistics can be found at <http://www.google.com/ads/innovations/ace.html>

guarantees subject randomness and is more transparent) and the context (language, network, etc.). Then we activate the ads and wait. As soon as data begins to be collected we can monitor the two conditions according to:

- **Basic Metrics:** the highest CTR measure indicates which message is best performing. It indicates which message has the highest initial impact.
- **Google Analytics Metrics:** measures how much the messages kept subjects on the site and how many pages have been viewed. Indicates interest/attitude generated in the subjects.
- **Conversion Metrics:** measures how much the messages converted subjects to the final goal. Indicates complete success of the persuasive message.
- **ROI Metrics:** by creating specific ROI values for every action the user performs on the landing page. The more relevant (from a persuasive point of view) the action the user performs, the higher the value we must assign to that action. In our view combined measurements are better: for example, there could be cases of messages with a lower CTR but a higher conversion rate.

Furthermore, AdWords allows very complex targeting options that can help in many different evaluation scenarios:

- **Language** (see how message impact can vary in different languages).
- **Location** (see how message impact can vary in different cultures sharing the same language).
- **Keyword matching** (see how message impact can vary with users having different interests).
- **Placements** (see how message impact can vary among people having different values - e.g. the same message displayed on Democrat or Republican web sites).
- **Demographics** (see how message impact can vary according to user gender and age).

### 5.1 Setting up an Experiment

To test the extent to which AdWords can be exploited, we focused on how to evaluate lexical variations of a message. In particular we were interested

in gaining insights about a system for *affective* variations of existing commentaries on medieval frescoes for a mobile museum guide that attracts the attention of visitors towards specific paintings (Guerini et al., 2008; Guerini et al., 2011b). The various steps for setting up an experiment (or a series of experiments) are as follows:

**Choose a Partner.** If you have the opportunity to have a commercial partner that already has the infrastructure for experiments (website, products, etc.) many of the following steps can be skipped. We assume that this is not the case.

**Choose a scenario.** Since you may not be equipped with a VAT code (or with the commercial partner that furnishes the AdWords account and infrastructure), you may need to “invent something to promote” without any commercial aim. If a “social marketing” scenario is chosen you can select “personal” as a “tax status”, that do not require a VAT code. In our case we selected cultural heritage promotion, in particular the frescoes of Torre Aquila (“Eagle Tower”) in Trento. The tower contains a group of 11 frescoes named “Ciclo dei Mesi” (cycle of the months) that represent a unique example of non-religious medieval frescoes in Europe.

**Choose an appropriate keyword** on which to advertise, “medieval art” in our case. It is better to choose keywords with enough web traffic in order to speed up the experimental process. In our case the search volume for “medieval art” (in phrase match) was around 22.000 hits per month. Another suggestion is to restrict the matching modality on Keywords in order to have more control over the situations in which ads are displayed and to avoid possible extraneous effects (the order of control for matching modality is: *[exact match]*, *[phrase match]* and *[broad match]*).

Note that such a technical decision - which keyword to use - is better taken at an early stage of development because it affects the following steps.

**Write messages** optimized for that keyword (e.g. including it in the title or the body of the ad). Such optimization must be the same for control and experimental condition. The rest of the ad can be designed in such a way to meet control and experimental condition design (in our case a message with slightly affective terms and a similar message with more affectively loaded variations)

**Build an appropriate landing page**, according to the keyword and populate the website pages with relevant material. This is necessary to create a “credible environment” for users to interact with.

**Incorporate meaningful actions in the website.** Users can perform various actions on a site, and they can be monitored. The design should include actions that are meaningful indicators of persuasive effect/success of the message. In our case we decided to include some outbound links, representing:

- general interest: “Buonconsiglio Castle site”
- specific interest: “Eagle Tower history”
- activated action: “Timetable and venue”
- complete success: “Book a visit”

Furthermore, through new Google Analytics features, we set up a series of *time\_spent\_on\_site* and *number\_of\_visited\_pages* thresholds to be monitored in the ACE tool.

## 5.2 Tips for Planning an Experiment

There are variables, inherent in the Google AdWords mechanism, that from a research point of view we shall consider “extraneous”. We now propose tips for controlling such extraneous variables.

**Add negative matching Keywords:** To add more control, if in doubt, put the words/expressions of the control and experimental conditions as negative keywords. This will prevent different highlighting between the two conditions that can bias the results. It is not strictly necessary since one can always control which queries triggered a click through the report menu. An example: if the only difference between control and experimental condition is the use of the adjectives “*gentle* knights” vs. “*valorous* knights”, one can use two negative keyword matches: *-gentle* and *-valorous*. Obviously if you are using a keyword in exact matching to trigger your ads, such as *[knight]*, this is not necessary.

**Frequency capping for the display network:** if you are running ads on the display network, you can use the “frequency capping” option set to 1 to add more control to the experiment. In this way it is assured that ads are displayed only one time per user on the display network.

**Placement bids for the search network:** unfortunately this option is no longer available. Basically the option allowed to bid only for certain positions

on the SERPs to avoid possible “extraneous variables effect” given by the position. This is best explained via an example: if, for whatever reason, one of the two ads gets repeatedly promoted to the premium position on the SERPs, then the CTR difference between ads would be strongly biased. From a research point of view “premium position” would then be an extraneous variable to be controlled (i.e. either both ads get an equal amount of premium position impressions, or both ads get no premium position at all). Otherwise the difference in CTR is determined by the “premium position” rather than by the independent variable under investigation (presence/absence of particular affective terms in the text ad). However even if it is not possible to rule out this “position effect” it is possible to monitor it by using the report (*Segment > Top vs. other + Experiment*) and checking how many times each ad appeared in a given position on the SERPs, and see if the ACE tool reports any statistical difference in the frequencies of ads positions.

**Extra experimental time:** While planning an experiment, you should also take into account the ads reviewing time that can take up to several days, in worst case scenarios. Note that when ads are in *eligible* status, they begin to show on the Google Network, but they are not approved yet. This means that the ads can only run on Google search pages and can only show for users who have turned off SafeSearch filtering, until they are approved. Eligible ads cannot run on the Display Network. This status will provide much less impressions than the final “approved” status.

**Avoid seasonal periods:** for the above reason, and to avoid extra costs due to high competition, avoid seasonal periods (e.g. Christmas time).

**Delivery method:** if you are planning to use the Accelerated Delivery method in order to get the results as quick as possible (in the case of “quick and dirty” experiments or “fast prototyping-evaluation cycles”) you should consider monitoring your experiment more often (even several times per day) to avoid running out of budget during the day.

## 6 Experiments

We ran two pilot experiments to test how affective variations of existing texts alter their persuasive im-

pact. In particular we were interested in gaining initial insights about an intelligent system for affective variations of existing commentaries on medieval frescoes.

We focused on adjective variations, using a slightly biased adjective for the control conditions and a strongly biased variation for the experimental condition. In these experiments we took it for granted that affective variations of a message work better than a neutral version (Van Der Sluis and Mellish, 2010), and we wanted to explore more finely grained tactics that involve the *grade of the variation* (i.e. a moderately positive variation vs. an extremely positive variation). Note that this is a more difficult task than the one proposed in (Van Der Sluis and Mellish, 2010), where they were testing long messages with lots of variations and with polarized conditions, neutral vs. biased. In addition we wanted to test how quickly experiments could be performed (two days versus the two week suggestion of Google).

Adjectives were chosen according to MAX bigram frequencies with the modified noun, using the Web 1T 5-gram corpus (Brants and Franz, 2006). Deciding whether this is the best metric for choosing adjectives to modify a noun or not (e.g. also pointwise mutual-information score can be used with a different rationale) is out of the scope of the present paper, but previous work has already used this approach (Whitehead and Cavedon, 2010). Top ranked adjectives were then manually ordered - according to affective weight - to choose the best one (we used a standard procedure using 3 annotators and a reconciliation phase for the final decision).

### 6.1 First Experiment

The first experiment lasted 48 hour with a total of 38 thousand subjects and a cost of 30 euros (see Table 1 for the complete description of the experimental setup). It was meant to test *broadly* how affective variations in the body of the ads performed. The two variations contained a fragment of a commentary of the museum guide; the control condition contained “*gentle knight*” and “*African lion*”, while in the experimental condition the affective loaded variations were “*valorous knight*” and “*indomitable lion*” (see Figure 1, for the complete ads). As can be seen from Table 2, the experiment did not yield any significant

result, if one looks at the overall analysis. But segmenting the results according to the keyword that triggered the ads (see Table 3) we discovered that on the “medieval art” keyword, the control condition performed better than the experimental one.

---

<b>Starting Date:</b> 1/2/2012
<b>Ending Date:</b> 1/4/2012
<b>Total Time:</b> 48 hours
<b>Total Cost:</b> 30 euros
<b>Subjects:</b> 38,082
<b>Network:</b> Search and Display
<b>Language:</b> English
<b>Locations:</b> Australia; Canada; UK; US
<b>KeyWords:</b> “medieval art”, pictures middle ages

---

Table 1: First Experiment Setup

ACE split	Clicks	Impr.	CTR
Control	31	18,463	0.17%
Experiment	20	19,619	0.10%
Network	Clicks	Impr.	CTR
Search	39	4,348	0.90%
Display	12	34,027	0.04%
TOTAL	51	38,082	0.13%

Table 2: First Experiment Results

Keyword	ACE split	Impr.	CTR
”medieval art”	Control	657	0.76%
”medieval art”	Experiment	701	<b>0.14%*</b>
medieval times history	Control	239	1.67%
medieval times history	Experiment	233	0.86%
pictures middle ages	Control	1114	1.35%
pictures middle ages	Experiment	1215	0.99%

Table 3: First Experiment Results Detail. \* indicates a statistically significant difference with  $\alpha < 0.01$

**Discussion.** As already discussed, user motivation is a key element for success in such fine-grained experiments: while less focused keywords did not yield any statistically significant differences, the most specialized keyword “medieval art” was the one that yielded results (i.e. if we display messages like those in Figure 1, that are concerned with medieval art frescoes, only those users really interested in the topic show different reaction patterns to the affective variations, while those generically interested in medieval times behave similarly in the two conditions). In the following experiment we tried to see

whether such variations have different effects when modifying a different element in the text.

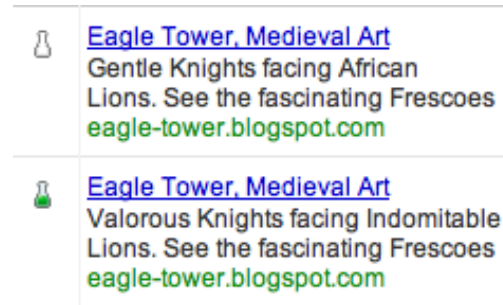


Figure 1: Ads used in the first experiment

## 6.2 Second Experiment

The second experiment lasted 48 hours with a total of one thousand subjects and a cost of 17 euros (see Table 4 for the description of the experimental setup). It was meant to test *broadly* how affective variations introduced in the title of the text Ads performed. The two variations were the same as in the first experiment for the control condition “gentle knight”, and for the experimental condition “valorous knight” (see Figure 2 for the complete ads). As can be seen from Table 5, also in this case the experiment did not yield any significant result, if one looks at the overall analysis. But segmenting the results according to the search network that triggered the ads (see Table 6) we discovered that on the search partners at the “other” position, the control condition performed better than the experimental one. Unlike the first experiment, in this case we segmented according to the ad position and search network typology since we were running our experiment only on one keyword in exact match.

---

<b>Starting Date:</b> 1/7/2012
<b>Ending Date:</b> 1/9/2012
<b>Total Time:</b> 48 hours
<b>Total Cost:</b> 17.5 euros
<b>Subjects:</b> 986
<b>Network:</b> Search
<b>Language:</b> English
<b>Locations:</b> Australia; Canada; UK; US
<b>KeyWords:</b> [medieval knights]

---

Table 4: Second Experiment Setup



Figure 2: Ads used in the second experiment

ACE split	Clicks	Impr.	CTR
Control	10	462	2.16%
Experiment	8	<b>524<sup>†</sup></b>	1.52%
TOTAL	18	986	1.82%

Table 5: Second Experiment Results. <sup>†</sup> indicates a statistically significant difference with  $\alpha < 0.05$

Top vs. Other	ACE split	Impr.	CTR
Google search: Top	Control	77	6.49%
Google search: Top	Experiment	68	2.94%
Google search: Other	Control	219	0.00%
Google search: Other	Experiment	<b>277*</b>	0.36%
Search partners: Top	Control	55	3.64%
Search partners: Top	Experiment	65	6.15%
Search partners: Other	Control	96	3.12%
Search partners: Other	Experiment	105	<b>0.95%<sup>†</sup></b>
Total - Search	-	986	1.82%

Table 6: Second Experiment Results Detail. <sup>†</sup> indicates a statistical significance with  $\alpha < 0.05$ , \* indicates a statistical significance with  $\alpha < 0.01$

**Discussion.** From this experiment we can confirm that at least under some circumstances a mild affective variation performs better than a strong variation. This mild variations seems to work better when user attention is high (the difference emerged when ads are displayed in a non-prominent position). Furthermore it seems that modifying the title of the ad rather than the content yields better results: 0.9% vs. 1.83% CTR ( $\chi^2 = 6.24$ ; 1 degree of freedom;  $\alpha < 0,01$ ) even if these results require further assessment with dedicated experiments.

As a side note, in this experiment we can see the problem of extraneous variables: according to AdWords' internal mechanisms, the experimental condition was displayed more often in the Google

search Network on the “other” position (277 vs. 219 impressions - and overall 524 vs. 462), still from a research perspective this is not a interesting statistical difference, and ideally should not be present (i.e. ads should get an equal amount of impressions for each position).

## Conclusions and future work

AdWords gives us an appropriate context for evaluating persuasive messages. The advantages are fast experiment building and evaluation, fully-automated analysis, and low cost. By using keywords with a low CPC it is possible to run large-scale experiments for just a few euros. AdWords proved to be very accurate, flexible and fast, far beyond our expectations. We believe careful design of experiments will yield important results, which was unthinkable before this opportunity for studies on persuasion appeared.

The motivation for this work was exploration of the impact of short persuasive messages, so to assess the effectiveness of different linguistic choices. The experiments reported in this paper are illustrative examples of the method proposed and are concerned with the evaluation of the role of minimal affective variations of short expressions. But there is enormous further potential in the proposed approach to ecological crowdsourcing for NLP: for instance, different rhetorical techniques can be checked in practice with large audiences and fast feedback. The assessment of the effectiveness of a change in the title as opposed to the initial portion of the text body provides a useful indication: one can investigate if variations inside the *given* or the *new* part of an expression or in the *topic* vs. *comment* (Levinson, 1983) have different effects. We believe there is potential for a concrete extensive exploration of different linguistic theories in a way that was simply not realistic before.

## Acknowledgments

We would like to thank Enrique Alfonseca and Steve Barrett, from Google Labs, for valuable hints and discussion on AdWords features. The present work was partially supported by a Google Research Award.

## References

- P. Andrews, S. Manandhar, and M. De Boni. 2008. Argumentative human computer dialogue for automated persuasion. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue*, pages 138–147. Association for Computational Linguistics.
- S. Aral and D. Walker. 2010. Creating social contagion through viral product design: A randomized trial of peer influence in networks. In *Proceedings of the 31th Annual International Conference on Information Systems*.
- T. Brants and A. Franz. 2006. Web 1t 5-gram corpus version 1.1. *Linguistic Data Consortium*.
- BJ Fogg. 2009. Creating persuasive technologies: An eight-step design process. *Proceedings of the 4th International Conference on Persuasive Technology*.
- M. Guerini, O. Stock, and C. Strapparava. 2008. Valentino: A tool for valence shifting of natural language texts. In *Proceedings of LREC 2008*, Marrakech, Morocco.
- M. Guerini, C. Strapparava, and O. Stock. 2010. Evaluation metrics for persuasive nlp with google adwords. In *Proceedings of LREC-2010*.
- M. Guerini, O. Stock, M. Zancanaro, D.J. O’Keefe, I. Mazzotta, F. Rosis, I. Poggi, M.Y. Lim, and R. Aylett. 2011a. Approaches to verbal persuasion in intelligent user interfaces. *Emotion-Oriented Systems*, pages 559–584.
- M. Guerini, C. Strapparava, and O. Stock. 2011b. Slanting existing text with Valentino. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 439–440. ACM.
- S.C. Levinson. 1983. *Pragmatics*. Cambridge University Press.
- W. Mason and S. Suri. 2010. Conducting behavioral research on amazon’s mechanical turk. *Behavior Research Methods*, pages 1–23.
- M. Negri, L. Bentivogli, Y. Mehdad, D. Giampiccolo, and A. Marchetti. 2010. Divide and conquer: Crowdsourcing the creation of cross-lingual textual entailment corpora. *Proc. of EMNLP 2011*.
- E. Reiter, R. Robertson, and L. Osman. 2003. Lesson from a failure: Generating tailored smoking cessation letters. *Artificial Intelligence*, 144:41–58.
- R. Snow, B. O’Connor, D. Jurafsky, and A.Y. Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 254–263. Association for Computational Linguistics.
- I. Van Der Sluis and C. Mellish. 2010. Towards empirical evaluation of affective tactical nlg. In *Empirical methods in natural language generation*, pages 242–263. Springer-Verlag.
- S. Whitehead and L. Cavedon. 2010. Generating shifting sentiment for a conversational agent. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, pages 89–97, Los Angeles, CA, June. Association for Computational Linguistics.



# Polarity Consistency Checking for Sentiment Dictionaries

**Eduard Dragut**   **Hong Wang**   **Clement Yu**   **Prasad Sistla**   **Weiwei Meng**  
Cyber Center                      Computer Science Dept.                      Computer Science Dept.  
Purdue University                      University of Illinois at Chicago                      Binghamton University  
edragut@purdue.edu                      {hwang207, cyu, sistla}@uic.edu                      meng@cs.binghamton.edu

## Abstract

Polarity classification of words is important for applications such as Opinion Mining and Sentiment Analysis. A number of sentiment word/sense dictionaries have been manually or (semi)automatically constructed. The dictionaries have substantial inaccuracies. Besides obvious instances, where the same word appears with different polarities in different dictionaries, the dictionaries exhibit complex cases, which cannot be detected by mere manual inspection. We introduce the concept of polarity consistency of words/senses in sentiment dictionaries in this paper. We show that the consistency problem is NP-complete. We reduce the polarity consistency problem to the satisfiability problem and utilize a fast SAT solver to detect inconsistencies in a sentiment dictionary. We perform experiments on four sentiment dictionaries and WordNet.

## 1 Introduction

The opinions expressed in various Web and media outlets (e.g., blogs, newspapers) are an important yardstick for the success of a product or a government policy. For instance, a product with consistently good reviews is likely to sell well. The general approach is to summarize the *semantic polarity* (i.e., positive or negative) of sentences/documents by analysis of the orientations of the individual words (Pang and Lee, 2004; Danescu-N.-M. et al., 2009; Kim and Hovy, 2004; Takamura et al., 2005). Sentiment dictionaries are utilized to facilitate the summarization. There are numerous works that, given a sentiment lexicon, analyze the structure of

a sentence/document to infer its orientation, the holder of an opinion, the sentiment of the opinion, etc. (Breck et al., 2007; Ding and Liu, 2010; Kim and Hovy, 2004). Several domain independent sentiment dictionaries have been manually or (semi)-automatically created, e.g., General Inquirer (GI) (Stone et al., 1996), Opinion Finder (OF) (Wilson et al., 2005), Appraisal Lexicon (AL) (Taboada and Grieve, 2004), SentiWordNet (Baccianella et al., 2010) and Q-WordNet (Agerri and García-Serrano, 2010). Q-WordNet and SentiWordNet are lexical resources which classify the synsets(senses) in WordNet according to their polarities. We call them *sentiment sense dictionaries* (SSD). OF, GI and AL are called *sentiment word dictionaries* (SWD). They consist of words manually annotated with their corresponding polarities. The sentiment dictionaries have the following problems:

- They exhibit substantial (intra-dictionary) inaccuracies. For example, the synset  $\{\textit{Indo-European}, \textit{Indo-Aryan}, \textit{Aryan}\}$  (*of or relating to the former Indo-European people*), has a negative polarity in Q-WordNet, while most people would agree that this synset has a neutral polarity instead.
- They have (inter-dictionary) inconsistencies. For example, the adjective `cheap` is positive in AL and negative in OF.
- These dictionaries do not address the concept of polarity (in)consistency of words/synsets.

We concentrate on the concept of (in)consistency in this paper. We define consistency among the polarities of words/synsets in a dictionary and give methods to check it. A couple of examples help illustrate the problem we attempt to address.

The first example is the verbs `confute` and `disprove`, which have positive and negative polarities, respectively, in OF. According to WordNet, both words have a unique sense, which they share: *disprove, confute (prove to be false) "The physicist disproved his colleagues' theories"*

Assuming that WordNet has complete information about the two words, it is rather strange that the words have distinct polarities. By manually checking two other authoritative English dictionaries, Oxford<sup>1</sup> and Cambridge<sup>2</sup>, we note that the information about `confute` and `disprove` in WordNet is the same as that in these dictionaries. So, the problem seems to originate in OF.

The second example is the verbs `tantalize` and `taunt`, which have positive and negative polarities, respectively, in OF. They also have a unique sense in WordNet, which they share. Again, there is a contradiction. In this case Oxford dictionary mentions a sense of `tantalize` that is missing from WordNet: *"excite the senses or desires of (someone)"*. This sense conveys a positive polarity. Hence, `tantalize` conveys a positive sentiment when used with this sense.

In summary, these dictionaries have conflicting information. Manual checking of sentiment dictionaries for inconsistency is a difficult endeavor. We deem words such as `confute` and `disprove` inconsistent. We aim to unearth these inconsistencies in sentiment dictionaries. The presence of inconsistencies found via polarity analysis is not exclusively attributed to one party, i.e., either the sentiment dictionary or WordNet. Instead, as emphasized by the above examples, some of them lie in the sentiment dictionaries, while others lie in WordNet. Therefore, a by-product of our polarity consistency analysis is that it can also locate some of the likely places where WordNet needs linguists' attention.

We show that the problem of checking whether the polarities of a set of words is consistent is NP-complete. Fortunately, the consistency problem can be reduced to the satisfiability problem (SAT). A fast SAT solver is utilized to detect inconsistencies and it is known such solvers can in practice determine consistency or detect inconsistencies. Experimental results show that substantial inconsistencies

are discovered among words with polarities within and across sentiment dictionaries. This suggests that some remedial work needs to be performed on these sentiment dictionaries as well as on WordNet. The contributions of this paper are:

- address the consistency of polarities of words/senses. The problem has not been addressed before;
- show that the consistency problem is NP-complete;
- reduce the polarity consistency problem to the satisfiability problem and utilize a fast SAT solver to detect inconsistencies;
- give experimental results to demonstrate that our technique identifies considerable inconsistencies in various sentiment lexicons as well as discrepancies between these lexicons and WordNet.

## 2 Problem Definition

The polarities of the words in a sentiment dictionary may not necessarily be consistent (or correct). In this paper, we focus on the detection of polarity assignment inconsistencies for the words and synsets within and across dictionaries (e.g., OF vs. GI). We attempt to pinpoint the words with polarity inconsistencies and classify them (Section 3).

### 2.1 WordNet

We give a formal characterization of WordNet. This consists of words, synsets and frequency counts. A **word-synset network**  $\mathcal{N}$  is quadruple  $(\mathcal{W}, \mathcal{S}, \mathcal{E}, f)$  where  $\mathcal{W}$  is a finite set of words,  $\mathcal{S}$  is a finite set of synsets,  $\mathcal{E}$  is a set of undirected edges between elements in  $\mathcal{W}$  and  $\mathcal{S}$ , i.e.,  $\mathcal{E} \subseteq \mathcal{W} \times \mathcal{S}$  and  $f$  is a function assigning a positive integer to each element in  $\mathcal{E}$ . For an edge  $(w, s)$ ,  $f(w, s)$  is called the **frequency of use** of  $w$  in the sense given by  $s$ . For any word  $w$  and synset  $s$ , we say that  $s$  is a synset of  $w$  if  $(w, s) \in \mathcal{E}$ . Also, for any word  $w$ , we let  $freq(w)$  denote the sum of all  $f(w, s)$  such that  $(w, s) \in \mathcal{E}$ . If a synset has a 0 frequency of use we replace it with 0.1, which is a standard smoothing technique (Han, 2005). For instance, the word `cheap` has four senses. The frequencies of occurrence of the word in the four senses are  $f_1 = 9$ ,  $f_2 = 1$ ,  $f_3 = 1$  and  $f_4 = 0$ , respectively. By smoothing,  $f_4 = 0.1$ . Hence,  $freq(cheap) = f_1 + f_2 + f_3 + f_4 = 11.1$ . The **relative frequency** of the synset in the first sense of `cheap`, which denotes the probability that the word is used in the first sense, is  $\frac{f_1}{freq(cheap)} = \frac{9}{11.1} = 0.81$ .

<sup>1</sup><http://oxforddictionaries.com/>

<sup>2</sup><http://dictionary.cambridge.org/>

## 2.2 Consistent Polarity Assignment

We assume that each synset has a *unique* polarity. We define the polarity of a word to be a discrete probability distribution:  $P_+, P_-, P_0$  with  $P_+ + P_- + P_0 = 1$ , where they represent the “likelihoods” that the word is positive, negative or neutral, respectively. We call this distribution a **polarity distribution**. For instance, the word `cheap` has the polarity distribution  $P_+ = 0.81, P_- = 0.19$  and  $P_0 = 0$ . The polarity distribution of a word is estimated using the polarities of its underlying synsets. For instance `cheap` has four senses, with the first sense being positive and the last three senses being negative. The probability that the word expresses a negative sentiment is  $P_- = \frac{f_2 + f_3 + f_4}{\text{freq}(\text{cheap})} = 0.19$ , while the probability that the word expresses a positive sentiment is  $P_+ = \frac{f_1}{\text{freq}(\text{cheap})} = 0.81$ .  $P_0 = 1 - P_+ - P_- = 0$ .

Our view of characterizing the polarity of a word using a polarity distribution is shared with other previous works (Kim and Hovy, 2006; Andreevskaja and Bergler, 2006). Nonetheless, we depart from these works in the following key aspect. We say that a word has a (mostly) positive (negative) polarity if the **majority sense of the word** is positive (negative). That is, a word has a mostly positive polarity if  $P_+ > P_- + P_0$  and it has a mostly negative polarity if  $P_- > P_+ + P_0$ . Or, equivalently, if  $P_+ > \frac{1}{2}$  or  $P_- > \frac{1}{2}$ , respectively. For example, on majority, `cheap` conveys positive polarity since  $P_+ = .081 > \frac{1}{2}$ , i.e., the majority sense of the word `cheap` has positive connotation.

Based on this study, we contend that GI, OF and AL tacitly assume this property. For example, the verb `steal` is assigned only negative polarity in GI. This word has two other less frequently occurring senses, which have positive polarities. The polarity of `steal` according to these two senses is not mentioned in GI. This is the case for the overwhelming majority of the entries in the three dictionaries: only 112 out of a total of 14,105 entries in the three dictionaries regard words with multiple polarities. For example, the verb `arrest` is mentioned with both negative and positive polarities in GI. We regard an entry in an SWD as the majority sense of the word has the specified polarity, although the word may carry other polarities. For instance, the adjective `cheap` has positive polarity in GI. The only assumption we make about the word is that it has a po-

larity distribution such that  $P_+ > P_- + P_0$ . This interpretation is consistent with the senses of the word. In this work we show that this property allows the polarities of words in input sentiment dictionaries to be checked. We formally state this property.

**Definition 1.** *Let  $w$  be a word and  $S_w$  its set of synsets. Each synset in  $S_w$  has an associated polarity and a relative frequency with respect to  $w$ .  $w$  has polarity  $p$ ,  $p \in \{\text{positive, negative}\}$  if there is a subset of synsets  $S' \subseteq S_w$  such that each synset  $s \in S'$  has polarity  $p$  and  $\sum_{s \in S'} \frac{f(w,s)}{\text{freq}(w)} > 0.5$ .  $S'$  is called a **polarity dominant subset**. If there is no such subset then  $w$  has a neutral polarity.*

*$S' \subseteq S_w$  is a **minimally dominant subset of synsets (MDSs)** if the sum of the relative frequencies of the synsets in  $S'$  is larger than 0.5 and the removal of any synset  $s$  from  $S'$  will make the sum of the relative frequencies of the synsets in  $S' - \{s\}$  smaller than or equal to 0.5.*

The definition does not preclude a word from having a polarity with a *majority* sense and a different polarity with a *minority* sense. For example, the definition does not prevent a word from having both positive and negative senses, but it prevents a word from concomitantly having a majority sense of being positive and a majority sense of being negative.

Despite using a “hard-coded” constant in the definition, our approach is generic and does not depend on the constant 0.5. This constant is just a lower bound for deciding whether a word has a majority sense with a certain polarity. It also is intuitively appealing. The constant can be replaced with an arbitrary threshold  $\tau$  between 0.5 and 1.

We need a formal description of polarity assignments to the words and synsets in WordNet. We assign polarities from the set  $\mathcal{P} = \{\text{positive, negative, neutral}\}$  to elements in  $\mathcal{W} \cup \mathcal{S}$ . Formally, a polarity assignment  $\gamma$  for a network  $\mathcal{N}$  is a function from  $\mathcal{W} \cup \mathcal{S}$  to the set  $\mathcal{P}$ . Let  $\gamma$  be a polarity assignment for  $\mathcal{N}$ . We say that  $\gamma$  is **consistent** if it satisfies the following condition for each  $w \in \mathcal{W}$ :

For  $p \in \{\text{positive, negative}\}$ ,  $\gamma(w) = p$  iff the sum of all  $f(w, s)$  such that  $(w, s) \in \mathcal{E}$  and  $\gamma(s) = p$ , is greater than  $\frac{\text{freq}(w)}{2}$ . Note that, for any  $w \in \mathcal{W}$ ,  $\gamma(w) = \text{neutral}$  iff the above inequality is not satisfied for both values of  $p$  in  $\{\text{positive, negative}\}$ .

We contend that our approach is applicable to do-

Table 1: Disagreement between dictionaries.

Pairs of Dictionaries	Word Polarity Disagreement	
	Inconsistency	Overlap
OF & GI	90	2,924
OF & AL	73	1,150
GI & AL	18	712

main dependent sentiment dictionaries, too. We can employ WordNet Domains (Bentivogli et al., 2004). WordNet Domains augments WordNet with domain labels. Hence, we can project the words/synsets in WordNet according to a domain label and then apply our methodology to the projection.

### 3 Inconsistency Classification

Polarity inconsistencies are of two types: input and complex. We discuss them in this section.

#### 3.1 Input Dictionaries Polarity Inconsistency

Input polarity inconsistencies are of two types: intra-dictionary and inter-dictionary inconsistencies. The latter are obtained by comparing (1) two SWDs, (2) an SWD with an SSD and (3) two SSDs.

##### 3.1.1 Intra-dictionary inconsistency

An SWD may have triplets of the form  $(w, pos, p)$  and  $(w, pos, p')$ , where  $p \neq p'$ . For instance, the verb `brag` has both positive and negative polarities in OF. For these cases, we look up WordNet and apply Definition 1 to determine the polarity of word  $w$  with part of speech  $pos$ . The verb `brag` has negative polarity according to Definition 1. Such cases simply say that the team who constructs the dictionary believes the word has multiple polarities as they do not adopt our dominant sense principle. There are 58 occurrences of this type of inconsistency in GI, OF and AL. Q-WordNet, a sentiment sense dictionary, does not have intra-inconsistencies as it does not have a synset with multiple polarities.

##### 3.1.2 Inter-dictionary inconsistency

A word belongs to this category if it appears with different polarities in different SWDs. For instance, the adjective `joyless` has positive polarity in OF and negative polarity in GI. Table 1 depicts the overlapping relationships between the three SWDs: e.g., OF has 2,933 words in common with GI. The three dictionaries largely agree on the polarities of the words they pairwise share. For instance, out of 2,924 words shared by OF and GI, 2,834 have the same polarities. However, there are also a significant number

of words which have different polarities across dictionaries. Case in point, OF and GI disagree on the polarities of 90 words. Among the three dictionaries there are 181 polarity inconsistent words. These words are manually corrected using Definition 1 before the polarity consistency checking is applied to the union of the three dictionaries. This union is called *disagreement-free union*.

#### 3.2 Complex Polarity Inconsistency

This kind of inconsistency is more subtle and cannot be detected by direct comparison of words/synsets. They consist of *sets* of words and/or synsets whose polarities cannot concomitantly be satisfied. Recall the example of the verbs `confute` and `disprove` in OF given in Section 1. Recall our argument that by assuming that WordNet is correct, it is not possible for the two words to have different polarities: the sole synset, which they share, would have two different polarities, which is a contradiction.

The occurrence of an inconsistency points out the presence of incorrect input data:

- the information given in WordNet is incorrect, or
- the information in the given sentiment dictionary is incorrect, or both.

Regarding WordNet, the errors may be due to (1) a word has senses that are missing from WordNet or (2) the frequency count of a synset is inaccurate. A comprehensive analysis of every synset/word with inconsistency is a tantalizing endeavor requiring not only a careful study of multiple sources (e.g., dictionaries such as Oxford and Cambridge) but also linguistic expertise. It is beyond the scope of this paper to enlist all potentially inconsistent words/synsets and the possible remedies. Instead, we limit ourselves to drawing attention to the occurrence of these issues through examples, welcoming experts in the area to join the corrective efforts. We give more examples of inconsistencies in order to illustrate additional discrepancies between input dictionaries.

##### 3.2.1 WordNet vs. Sentiment Dictionaries

The adjective `bully` is an example of a discrepancy between WordNet and a sentiment dictionary. The word has negative polarity in OF and has a single sense in WordNet. The sense is shared with the word `nifty`, which has positive polarity in OF. By applying Definition 1 to `nifty` we obtain that the sense is positive, which in turn, by Definition 1, implies that `bully` is positive. This contradicts the

input polarity of `bully`. According to the Webster dictionary, the word has a sense (i.e., *resembling or characteristic of a bully*) which has a negative polarity, but it is not present in WordNet. The example shows the presence of a discrepancy between WordNet and OF, namely, OF seems to assign polarity to a word according to a sense that is not in WordNet.

### 3.2.2 Across Sentiment Dictionaries

We provide examples of inconsistencies across sentiment dictionaries here. Our first example is obtained by comparing SWDs. The adjective `comic` has negative polarity in AL and the adjective `laughable` has positive polarity in OF. Through deduction (i.e., by successive applications of Definition 1), the word `risible`, which is not present in either of the dictionaries, is assigned negative polarity because of `comic` and is assigned positive polarity because of `laughable`.

The second example illustrates that an SWD and an SSD may have contradicting information. The verb `intoxicate` has three synsets in WordNet, each with the same frequency. Hence, their relative frequencies with respect to `intoxicate` are  $\frac{1}{3}$ . On one hand, `intoxicate` has a negative polarity in GI. This means that  $P_- > \frac{1}{2}$ . On the other hand, two of its three synsets have positive polarity in Q-WordNet. So,  $P_+ = \frac{2}{3} > \frac{1}{2}$ , which means that  $P_- < \frac{1}{2}$ . This is a contradiction. This example can also be used to illustrate the presence of a discrepancy between WordNet and sentiment dictionaries. Note that all the frequencies of use of the senses of `intoxicate` in WordNet are 0. The problem is that when all the senses of a word have a 0 frequency of use, wrong polarity inference may be produced.

### 3.3 Consistent Polarity Assignment

Given the discussion above, it clearly is important to find all occurrences of inconsistent words. This in turn boils down to finding those words with the property that there does not exist any polarity assignment to the synsets, which is consistent with their polarities. It turns out that the complexity of the problem of assigning polarities to the synsets such that the assignment is consistent with the polarities of the input words, called `Consistent Polarity Assignment` problem, is a “hard” problem, as described below. The problem is stated as follows:

Consider two sets of nodes of type synsets and type words, in which each synset of a word has a

relative frequency with respect to the word. Each synset can be assigned a positive, negative or neutral polarity. A word has polarity  $p$  if it satisfies the hypothesis of Definition 1. The question to be answered is: Given an assignment of polarities to the words, does there exist an assignment of polarities to the synsets that agrees with that of the words?

In other words, given the polarities of a subset of words (e.g., that given by one of the three SWDs) the problem of finding the polarities of the synsets that agree with this assignment is a “hard” problem.

**Theorem 1.** *The Consistent Polarity Assignment problem is NP-complete.*

## 4 Polarity Consistency Checking

To “exhaustively” solve the problem of finding the polarity inconsistencies in an SWD, we propose a solution that reduces an instance of the problem to an instance of CNF-SAT. We can then employ a fast SAT solver (e.g., (Xu et al., 2008; Babic et al., 2006)) to solve our problem. CNF-SAT is a decision problem of determining if there is an assignment of True and False to the variables of a Boolean formula  $\Phi$  in conjunctive normal form (CNF) such that  $\Phi$  evaluates to True. A formula is in CNF if it is a conjunction of one or more clauses, each of which is a disjunction of literals. CNF-SAT is a classic NP-complete problem, but, modern SAT solvers are capable of solving many practical instances of the problem. Since, in general, there is no easy way to tell the difficulty of a problem without trying it, SAT solvers include time-outs, so they will terminate even if they cannot find a solution.

We developed a method of converting an instance of the polarity consistency checking problem into an instance of CNF-SAT, which we will describe next.

### 4.1 Conversion to CNF-SAT

The input consists of an SWD  $D$  and the word-synset network  $\mathcal{N}$ . We partition  $\mathcal{N}$  into connected components. For each synset  $s$  we define three Boolean variables  $s_-$ ,  $s_+$  and  $s_0$ , corresponding to the negative, positive and neutral polarities, respectively. In this section we use  $-$ ,  $+$ ,  $0$  to denote negative, positive and neutral polarities, respectively.

Let  $\Phi$  be the Boolean formula for a connected component  $M$  of the word-synset network  $\mathcal{N}$ . We introduce its clauses. First, for each synset  $s$  we need a clause  $C(s)$  that expresses that the synset can have

only one of the three polarities:  $C(s) = (s_+ \wedge \neg s_- \wedge \neg s_0) \vee (s_- \wedge \neg s_+ \wedge \neg s_0) \vee (s_0 \wedge \neg s_- \wedge \neg s_+)$ .

Since a word has a neutral polarity if it has neither positive nor negative polarities, we have that  $s_0 = \neg s_+ \wedge \neg s_-$ . Replacing this expression in the equation above and applying standard Boolean logic formulas, we can reduce it to

$$C(s) = \neg s_+ \vee \neg s_- \quad (1)$$

For each word  $w$  with polarity  $p \in \{-, +, 0\}$  in  $D$  we need a clause  $C(w, p)$  that states that  $w$  has polarity  $p$ . So, the Boolean formula for a connected component  $M$  of the word-synset network  $\mathcal{N}$  is:

$$\Phi = \bigwedge_{s \in M} C(s) \wedge \bigwedge_{(w,p) \in D} C(w,p). \quad (2)$$

From Definition 1,  $w$  is neutral if it is neither positive nor negative. Hence,  $C(w, 0) = \neg C(w, -) \wedge \neg C(w, +)$ . So, we need to define only the clauses  $C(w, -)$  and  $C(w, +)$ , which correspond to  $w$  having polarity negative and positive, respectively. So, herein  $p \in \{-, +\}$ , unless otherwise specified.

Our method is based on the following statement in Definition 1:  $w$  has polarity  $p$  if there exists a polarity dominant subset among its synsets. Thus,  $C(w, p)$  is defined by enumerating all the MDSs of  $w$ . If at least one of them is a polarity dominant subset then  $C(w, p)$  evaluates to True.

**Exhaustive Enumeration of MDSs Method (EEM)** We now elaborate the construction of  $C(w, p)$ . We enumerate all the MDSs of  $w$  and for each of them we introduce a clause. The clauses are then concatenated by OR in the Boolean formula. Let  $C(w, p, T)$  denote the clause for an MDS  $T$  of  $w$ , when  $w$  has polarity  $p \in \{-, +\}$ . Hence,

$$C(w, p) = \bigvee_{T \in MDS(w)} C(w, p, T), \quad (3)$$

where  $MDS(w)$  is the set of all MDSs of  $w$ .

For each MDS  $T$  of  $w$ , the clause  $C(w, p, T)$  is the AND of the variables corresponding to polarity  $p$  of the synsets in  $T$ . That is,

$$C(w, p, T) = \bigwedge_{s \in T} s_p, \quad p \in \{-, +\}. \quad (4)$$

The formula  $\Phi$  is not in CNF after this construction and it needs to be converted. The conversion to CNF is a standard procedure and we omit it in this paper.  $\Phi$  in CNF is input to a SAT solver.

**Example 1.** Consider a connected component consisting of the words  $w = \text{cheap}$ ,  $v = \text{inexpensive}$  and  $u = \text{sleazy}$ . *cheap* has a positive polarity, whereas *inexpensive* and *sleazy* have negative polarities. The synsets of these words are:  $\{s^1, s^2, s^3, s^4\}$ ,  $\{s^1\}$  and  $\{s^3, s^4, s^5\}$ , respectively (refer to WordNet). The relative frequencies of  $s^3$ ,  $s^4$  and  $s^5$  w.r.t. *sleazy* are all equal to 1/3. We have 15 binary variables, 3 per synset,  $s_-^i, s_+^i, s_0^i, 1 \leq i \leq 5$ . The only MDS of *cheap* is  $\{s^1\}$ , which coincides with that of *inexpensive*. Those of *sleazy* are  $\{s^3, s^4\}$ ,  $\{s^3, s^5\}$  and  $\{s^4, s^5\}$ . For each  $s^i$  we need a clause  $C(s^i)$ . Hence,  $C(w, +) = s_+^1$ ,  $C(v, -) = s_-^1$  and  $C(u, -) = (s_-^3 \wedge s_-^4) \vee (s_-^3 \wedge s_-^5) \vee (s_-^4 \wedge s_-^5)$ . Thus,  $\Phi = \bigwedge C(s^i) \wedge [s_+^1 \wedge s_-^1 \wedge ((s_-^3 \wedge s_-^4) \vee (s_-^3 \wedge s_-^5) \vee (s_-^4 \wedge s_-^5))]$ .  $\Phi$  is not in CNF and needs to be converted. For  $\Phi$  to be True, the clauses  $C(w, +) = s_+^1$  and  $C(v, -) = s_-^1$  must be True. But, this makes  $C(s^1)$  False. Hence,  $\Phi$  is not satisfiable. The clauses  $C(w, +) = s_+^1$  and  $C(v, -) = s_-^1$  are unsatisfiable and thus the polarities of *cheap* and *inexpensive* are inconsistent.

## 4.2 Implementation Issues

The above reduction is exponential in the number of clauses (see, Equation 3) in the worst case. A polynomial reduction is possible, but it is significantly more complicated to implement. We choose to present the exponential reduction in this paper because it can handle over 97% of the words in WordNet and it is better suited to explain one of the main contributions of paper: the translation from the polarity consistency problem to SAT.

WordNet possesses nice properties, which allows the exponential reduction to run efficiently in practice. First, 97.2% of its (word, part-of-speech) pairs have 4 or fewer synsets. Thus, these words add very few clauses to a CNF formula (Equation 3). Second, WordNet can be partitioned into 33,015 *non-trivial* connected components, each of which corresponds to a Boolean formula and they all are independently handled. A non-trivial connected component has at least two words. Finally, in practice, not all connected components need to be considered for an input sentiment dictionary  $D$ , but only those having at least two words in  $D$ . In our experiments the largest number of components that need to be processed is

Table 2: Distribution of words and synsets

POS	Words	Synsets	OF	GI	AL	QWN
<b>Noun</b>	117,798	82,115	1,907	1,444	2	7,403
<b>Verb</b>	11,529	13,767	1,501	1,041	0	4006
<b>Adj.</b>	21,479	18,156	2,608	1,188	1,440	4050
<b>Adv.</b>	4,481	3,621	775	51	317	40
<b>Total</b>	<b>155,287</b>	<b>117,659</b>	<b>6,791</b>	<b>3,961</b>	<b>1,759</b>	<b>15,499</b>

1,581, for the disagreement-free union dictionary.

## 5 Detecting Inconsistencies

In this section we describe how we detect the words with polarity inconsistencies using the output of a SAT solver. For an unsatisfiable formula, a modern SAT solver returns a *minimal unsatisfiable core* (MUC) from the original formula. An *unsatisfiable core* is minimal if it becomes satisfiable whenever any one of its clauses is removed. There are no known practical algorithms for computing the *minimum core* (Dershowitz et al., 2006). In our problem a MUC corresponds to a set of polarity inconsistent words. The argument is as follows. Consider  $W$  the set of words in a connected component and  $\Phi$  the CNF formula generated with the above method. During the transformation we keep track of the clauses introduced in  $\Phi$  by each word. Suppose  $\Phi$  is inconsistent. Then, the SAT solver returns a MUC. Each clause in a MUC is mapped back to its corresponding word(s). We obtain the corresponding subset of words  $W'$ ,  $W' \subseteq W$ . Suppose that  $\Phi'$  is the Boolean CNF formula for the words in  $W'$ . The set of clauses in  $\Phi'$  is a subset of those in  $\Phi$ . Also, the clauses in the MUC appear in  $\Phi'$ . Thus,  $\Phi'$  is unsatisfiable and the words in  $W'$  are inconsistent.

To find *all* inconsistent words we ought to generate all MUCs. Unfortunately, this is a “hard” problem (Dershowitz et al., 2006) and no open source SAT solver possesses this functionality. We however observe that the two SAT solvers we use for our experiments (SAT4j and PicoSAT (Biere, 2008)) return different MUCs for the same formula and we use them to find as many inconsistencies as possible.

## 6 Experiments

The goal of the experimental study is to show that our techniques can identify considerable inconsistencies in various sentiment dictionaries.

Table 3: Intra- and inter-dictionaries inconsistency

POS	OF	QW	GI	QW	AL	QW	UF	QW
<b>Noun</b>	23	119	4	61	0	42	90	140
<b>Verb</b>	66	113	2	67	0	0	63	137
<b>Adj.</b>	90	170	8	48	0	0	27	177
<b>Adv.</b>	61	1	0	0	2	0	69	1
<b>Total</b>	<b>240</b>	<b>403</b>	<b>14</b>	<b>176</b>	<b>2</b>	<b>42</b>	<b>249</b>	<b>455</b>

**Data sets** In our experiments, we use WordNet 3.0, GI, OF, AL and Q-WordNet. Their statistics are given in Table 2. The table shows the distribution of the words and synsets per part of speech. Columns 2 and 3 pertain to WordNet. There are 3,961 entries in GI, 1,759 entries in AL and 6,791 entries in OF which appear in WordNet. Q-WordNet has 15,499 entries, i.e., synsets with polarities.

**Inconsistency Detection** We applied our method to (1) each of AL, GI and OF; (2) the disagreement-free union (UF); (3) each of AL, GI and OF together with Q-WordNet and (4) UF and Q-WordNet. Table 3 summarizes the outcome of the experimental study. EEM finds 240, 14 and 2 polarity inconsistent words in OF, GI and AL, respectively. The ratio between the number of inconsistent words and the number of input words is the highest for OF and the lowest for AL. The union dictionary has 7,794 words and 249 out of them are found to be polarity inconsistent words. Recall that we manually corrected the polarities of 181 words, to the best of our understanding. So, in effect the three dictionaries have  $249 + 181 = 430$  polarity inconsistent words. As discussed in the previous section, these may not be all the polarity inconsistencies in UF. In general, to find all inconsistencies we need to generate all MUCs. Generating all MUCs is an “overkill” and the SAT solvers we use do not implement such a functionality. In addition, the intention of SAT solver designers is to use MUCs in an interactive manner. That is, the errors pointed out by a MUC are corrected and then the new improved formula is re-evaluated by the SAT solver. If an error is still present a new MUC is reported, and the process repeats until the formula has no errors. Or, in our problem, until a dictionary is consistent.

We also paired Q-WordNet with each of the SWDs. Table 3 presents the results. Observe that polarities assigned to the words in AL and GI largely agree with the polarities assigned to the synsets in

Q-WordNet. This is expected for AL because it has only two nouns and no verb, while Q-WordNet has only 40 adverbs. Consequently, these two dictionaries have limited “overlay”. The union dictionary and Q-WordNet have substantial inconsistencies: the polarity of 455 words in the union dictionary disagrees with the polarities assigned to their underlying synsets in Q-WordNet.

**Sentence Level Evaluation** We took 10 pairs of inconsistent words per part of speech; in total, we collected a set  $IW$  of 80 inconsistent words. Let  $\langle w, pos, p \rangle \in IW$ ,  $p$  is the polarity of  $w$ . We collected 5 sentences for  $\langle w, pos \rangle$  from the set of snippets returned by Google for query  $w$ . We parsed the snippets and identified the first 5 occurrences of  $w$  with the part of speech  $pos$ . Then two graduate students with English background analyzed the polarities of  $\langle w, pos \rangle$  in the 5 sentences. We counted the number of times  $\langle w, pos \rangle$  appears with polarity  $p$  and polarities different from  $p$ . We defined an agreement scale: total agreement (5/5), most agreement (4/5), majority agreement (3/5), majority disagreement (2/5), most disagreement (1/5), total disagreement (0/5). We computed the percentage of words per agreement category. We repeated the experiment for 40 randomly drawn words (10 per part of speech) from the set of consistent words. In total 600 sentences were manually analyzed. Figure 1 shows the distribution of the (in)consistent words. For example, the annotators totally agree with the polarities of 55% of the consistent words, whereas they only totally agree with 16% of the polarities of the inconsistent words. The graph suggests that the annotators disagree to some extent (total disagreement + most disagreement + major disagreement) with 40% of the polarities of the inconsistent words, whereas they disagree to some extent with only 5% of the consistent words. We also manually investigated the senses of these words in WordNet. We noted that 36 of the 80 inconsistent words (45%) have missing senses according to one of these English dictionaries: Oxford and Cambridge.

**Computational Issues** We used a 4-core CPU computer with 12GB of memory. EEM requires 10GB of memory and cannot handle words with more than 200,000 MDSs: for UF we left the SAT solver running for a week without ever terminating. In contrast, it takes about 4 hours if we limit the set

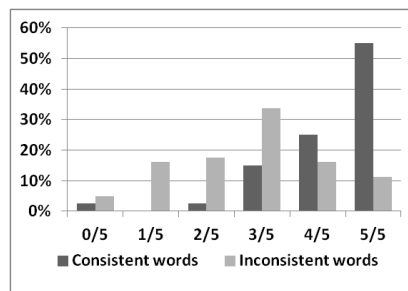


Figure 1: Human classification of (in)consistent words.

of words to those that have up to 200,000 MDSs. EEM could not handle words such as *make*, *give* and *break*. Recall however that we did not generate all MUCs. We do not know how long would that might have taken. (The polynomial method handles all the words in WordNet and it takes 5GB of memory and about 2 hours to finish.)

## 7 Related Work

Several researchers have studied the problem of finding opinion words (Liu, 2010). There are two lines of work on sentiment polarity lexicon induction: corpora-based (Hatzivassiloglou and McKeown, 1997; Kanayama and Nasukawa, 2006; Qiu et al., 2009; Wiebe, 2000) and dictionary-based (Andreevskaia and Bergler, 2006; Agerri and García-Serrano, 2010; Dragut et al., 2010; Esuli and Sebastiani, 2005; Baccianella et al., 2010; Hu and Liu, 2004; Kamps et al., 2004; Kim and Hovy, 2006; Rao and Ravichandran, 2009; Takamura et al., 2005). Our work falls into the latter. Most of these works use the lexical relations defined in WordNet (e.g., synonym, antonym) to derive sentiment lexicons. To our knowledge, none of the earlier works studied the problem of polarity consistency checking for a sentiment dictionary. Our techniques can pinpoint the inconsistencies within individual dictionaries and across dictionaries.

## 8 Conclusion

We studied the problem of checking polarity consistency for sentiment word dictionaries. We proved that this problem is NP-complete. We showed that in practice polarity inconsistencies of words both within a dictionary and across dictionaries can be obtained using an SAT solver. The inconsistencies are pinpointed and this allows the dictionaries to be improved. We reported experiments on four sentiment dictionaries and their union dictionary.



## Acknowledgments

This work is supported in part by the following NSF grants: IIS-0842546 and IIS-0842608.

## References

- Rodrigo Agerri and Ana García-Serrano. 2010. Q-wordnet: Extracting polarity from wordnet senses. In *LREC*.
- A. Andreevskaia and S. Bergler. 2006. Mining wordnet for fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *EACL*.
- Domagoj Babic, Jesse Bingham, and Alan J. Hu. 2006. B-cubing: New possibilities for efficient sat-solving. *TC*, 55(11).
- Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *LREC*, Valletta, Malta, May.
- Luisa Bentivogli, Pamela Forner, Bernardo Magnini, and Emanuele Pianta. 2004. Revising the wordnet domains hierarchy: semantics, coverage and balancing. *MLR*.
- Armin Biere. 2008. PicoSAT essentials. *JSAT*, 4(2-4):75–97.
- Eric Breck, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *IJCAI*.
- Cristian Danescu-N.-M., Gueorgi Kossinets, Jon Kleinberg, and Lillian Lee. 2009. How opinions are received by online communities: a case study on amazon.com helpfulness votes. In *WWW*, pages 141–150.
- Nachum Dershowitz, Ziyad Hanna, and Er Nadel. 2006. A scalable algorithm for minimal unsatisfiable core extraction. In *In Proc. SAT06*. Springer.
- Xiaowen Ding and Bing Liu. 2010. Resolving object and attribute coreference in opinion mining. In *COLING*.
- Eduard C. Dragut, Clement T. Yu, A. Prasad Sistla, and Weiyi Meng. 2010. Construction of a sentimental word dictionary. In *CIKM*, pages 1761–1764.
- Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. In *CIKM*, pages 617–624.
- Jiawei Han. 2005. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc.
- Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *ACL*, pages 174–181, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *ACM SIGKDD*, pages 168–177, New York, NY, USA. ACM.
- J. Kamps, M. Marx, R. Mokken, and M. de Rijke. 2004. Using wordnet to measure semantic orientation of adjectives. In *LREC*.
- Hiroshi Kanayama and Tetsuya Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06*, pages 355–363, Stroudsburg, PA, USA. Association for Computational Linguistics.
- M. Kim and E. Hovy. 2004. Determining the sentiment of opinions. In *COLING*.
- Soo-Min Kim and Eduard Hovy. 2006. Identifying and analyzing judgment opinions. In *HLT-NAACL*.
- Bing Liu. 2010. Sentiment analysis and subjectivity. In Nitin Indurkha and Fred J. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group, Boca Raton, FL. ISBN 978-1420085921.
- B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*.
- Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *IJCAI*, pages 1199–1204.
- Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *EACL*.
- P. Stone, D. Dunphy, M. Smith, and J. Ogilvie. 1996. The general inquirer: A computer approach to content analysis. In *MIT Press*.
- M. Taboada and J. Grieve. 2004. Analyzing appraisal automatically. In *AAAI Spring Symposium*.
- Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientations of words using spin model. In *ACL*, pages 133–140.
- Janyce Wiebe. 2000. Learning subjective adjectives from corpora. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 735–740. AAAI Press.
- T. Wilson, J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT/EMNLP*.
- Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. 2008. Satzilla: portfolio-based algorithm selection for sat. *J. Artif. Int. Res.*, 32:565–606, June.

# Combining Coherence Models and Machine Translation Evaluation Metrics for Summarization Evaluation

Ziheng Lin<sup>†</sup>, Chang Liu<sup>‡</sup>, Hwee Tou Ng<sup>‡</sup> and Min-Yen Kan<sup>‡</sup>

<sup>†</sup> SAP Research, SAP Asia Pte Ltd  
30 Pasir Panjang Road, Singapore 117440  
ziheng.lin@sap.com

<sup>‡</sup> Department of Computer Science, National University of Singapore  
13 Computing Drive, Singapore 117417  
{liuchan1, nght, kanmy}@comp.nus.edu.sg

## Abstract

An ideal summarization system should produce summaries that have high content coverage and linguistic quality. Many state-of-the-art summarization systems focus on content coverage by extracting content-dense sentences from source articles. A current research focus is to process these sentences so that they read fluently as a whole. The current AESOP task encourages research on evaluating summaries on content, readability, and overall responsiveness. In this work, we adapt a machine translation metric to measure content coverage, apply an enhanced discourse coherence model to evaluate summary readability, and combine both in a trained regression model to evaluate overall responsiveness. The results show significantly improved performance over AESOP 2011 submitted metrics.

## 1 Introduction

Research and development on automatic and manual evaluation of summarization systems have been mainly focused on content coverage (Lin and Hovy, 2003; Nenkova and Passonneau, 2004; Hovy et al., 2006; Zhou et al., 2006). However, users may still find it difficult to read such high-content coverage summaries as they lack fluency. To promote research on automatic evaluation of summary readability, the Text Analysis Conference (TAC) (Owczarzak and Dang, 2011) introduced a new subtask on readability to its Automatically Evaluating Summaries of Peers (AESOP) task.

Most of the state-of-the-art summarization systems (Ng et al., 2011; Zhang et al., 2011; Conroy et al., 2011) are extraction-based. They extract the most content-dense sentences from source articles. If no post-processing is performed to the generated summaries, the presentation of the extracted sentences may confuse readers. Knott (1996) argued that when the sentences of a text are randomly ordered, the text becomes difficult to understand, as its discourse structure is disturbed. Lin et al. (2011) validated this argument by using a trained model to differentiate an original text from a randomly-ordered permutation of its sentences by looking at their discourse structures. This prior work leads us to believe that we can apply such discourse models to evaluate the readability of extract-based summaries. We will discuss the application of Lin et al.'s discourse coherence model to evaluate readability of machine generated summaries. We also introduce two new feature sources to enhance the model with hierarchical and Explicit/Non-Explicit information, and demonstrate that they improve the original model.

There are parallels between evaluations of machine translation (MT) and summarization with respect to textual content. For instance, the widely used ROUGE (Lin and Hovy, 2003) metrics are influenced by BLEU (Papineni et al., 2002): both look at surface n-gram overlap for content coverage. Motivated by this, we will adapt a state-of-the-art, linear programming-based MT evaluation metric, TESLA (Liu et al., 2010), to evaluate the content coverage of summaries.

TAC's overall responsiveness metric evaluates the

quality of a summary with regard to both its content and readability. Given this, we combine our two component coherence and content models into an SVM-trained regression model as our surrogate to overall responsiveness. Our experiments show that the coherence model significantly outperforms all AESOP 2011 submissions on both initial and update tasks, while the adapted MT evaluation metric and the combined model significantly outperform all submissions on the initial task. To the best of our knowledge, this is the first work that applies a discourse coherence model to measure the readability of summaries in the AESOP task.

## 2 Related Work

Nenkova and Passonneau (2004) proposed a manual evaluation method that was based on the idea that there is no single best model summary for a collection of documents. Human annotators construct a *pyramid* to capture important Summarization Content Units (SCUs) and their weights, which is used to evaluate machine generated summaries.

Lin and Hovy (2003) introduced an automatic summarization evaluation metric, called ROUGE, which was motivated by the MT evaluation metric, BLEU (Papineni et al., 2002). It automatically determines the content quality of a summary by comparing it to the model summaries and counting the overlapping n-gram units. Two configurations – ROUGE-2, which counts bigram overlaps, and ROUGE-SU4, which counts unigram and bigram overlaps in a word window of four – have been found to correlate well with human evaluations.

Hovy et al. (2006) pointed out that automated methods such as ROUGE, which match fixed length n-grams, face two problems of tuning the appropriate fragment lengths and matching them properly. They introduced an evaluation method that makes use of small units of content, called Basic Elements (BEs). Their method automatically segments a text into BEs, matches similar BEs, and finally scores them.

Both ROUGE and BE have been implemented and included in the ROUGE/BE evaluation toolkit<sup>1</sup>, which has been used as the default evaluation tool in the summarization track in the Document Un-

derstanding Conference (DUC) and Text Analysis Conference (TAC). DUC and TAC also manually evaluated machine generated summaries by adopting the Pyramid method. Besides evaluating with ROUGE/BE and Pyramid, DUC and TAC also asked human judges to score every candidate summary with regard to its content, readability, and overall responsiveness.

DUC and TAC defined linguistic quality to cover several aspects: grammaticality, non-redundancy, referential clarity, focus, and structure/coherence. Recently, Pitler et al. (2010) conducted experiments on various metrics designed to capture these aspects. Their experimental results on DUC 2006 and 2007 show that grammaticality can be measured by a set of syntactic features, while the last three aspects are best evaluated by local coherence. Conroy and Dang (2008) combined two manual linguistic scores – grammaticality and focus – with various ROUGE/BE metrics, and showed this helps better predict the responsiveness of the summarizers.

Since 2009, TAC introduced the task of Automatically Evaluating Summaries of Peers (AESOP). AESOP 2009 and 2010 focused on two summary qualities: content and overall responsiveness. Summary content is measured by comparing the output of an automatic metric with the manual Pyramid score. Overall responsiveness measures a combination of content and linguistic quality. In AESOP 2011 (Owczarzak and Dang, 2011), automatic metrics are also evaluated for their ability to assess summary readability, *i.e.*, to measure how linguistically readable a machine generated summary is. Submitted metrics that perform consistently well on the three aspects include Giannakopoulos and Karkaletsis (2011), Conroy et al. (2011), and de Oliveira (2011). Giannakopoulos and Karkaletsis (2011) created two character-based n-gram graph representations for both the model and candidate summaries, and applied graph matching algorithm to assess their similarity. Conroy et al. (2011) extended the model in (Conroy and Dang, 2008) to include shallow linguistic features such as term overlap, redundancy, and term and sentence entropy. de Oliveira (2011) modeled the similarity between the model and candidate summaries as a maximum bipartite matching problem, where the two summaries are represented as two sets of nodes and precision and recall are cal-

<sup>1</sup><http://berouge.com/default.aspx>

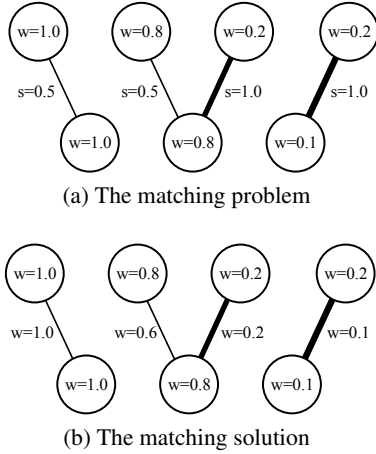


Figure 1: A BNG matching problem. Top and bottom rows of each figure represent BNG from the model and candidate summaries, respectively. Links are similarities. Both n-grams and links are weighted.

culated from the matched edges. However, none of the AESOP metrics currently apply deep linguistic analysis, which includes discourse analysis.

Motivated by the parallels between summarization and MT evaluation, we will adapt a state-of-the-art MT evaluation metric to measure summary content quality. To apply deep linguistic analysis, we also enhance an existing discourse coherence model to evaluate summary readability. We focus on metrics that measure the average quality of machine summarizers, *i.e.*, metrics that can rank a set of machine summarizers correctly (human summarizers are not included in the list).

### 3 TESLA-S: Evaluating Summary Content

TESLA (Liu et al., 2010) is an MT evaluation metric which extends BLEU by introducing a linear programming-based framework for improved matching. It also makes use of linguistic resources and considers both precision and recall.

#### 3.1 The Linear Programming Matching Framework

Figure 1 shows the matching of bags of n-grams (BNGs) that forms the core of the TESLA metric. The top row in Figure 1a represents the bag of n-grams (BNG) from the model summary, and the

bottom row represents the BNG from the candidate summary. Each n-gram has a weight. The links between the n-grams represent the similarity score, which are constrained to be between 0 and 1. Mathematically, TESLA takes as input the following:

1. The BNG of the model summary,  $X$ , and the BNG of the candidate summary,  $Y$ . The  $i$ th entry in  $X$  is  $x_i$  and has weight  $x_i^W$  (analogously for  $y_i$  and  $y_i^W$ ).
2. A similarity score  $s(x_i, y_j)$  between all n-grams  $x_i$  and  $y_j$ .

The goal of the matching process is to align the two BNGs so as to maximize the overall similarity. The variables of the problem are the allocated weights for the edges,

$$w(x_i, y_j) \quad \forall i, j$$

TESLA maximizes

$$\sum_{i,j} s(x_i, y_j) w(x_i, y_j)$$

subject to

$$\begin{aligned} w(x_i, y_j) &\geq 0 \quad \forall i, j \\ \sum_j w(x_i, y_j) &\leq x_i^W \quad \forall i \\ \sum_i w(x_i, y_j) &\leq y_j^W \quad \forall j \end{aligned}$$

This real-valued linear programming problem can be solved efficiently. The overall similarity  $S$  is the value of the objective function. Thus,

$$\begin{aligned} \text{Precision} &= \frac{S}{\sum_j y_j^W} \\ \text{Recall} &= \frac{S}{\sum_i x_i^W} \end{aligned}$$

The final TESLA score is given by the F-measure:

$$F = \frac{\text{Precision} \times \text{Recall}}{\alpha \times \text{Precision} + (1 - \alpha) \times \text{Recall}}$$

In this work, we set  $\alpha = 0.8$ , following (Liu et al., 2010). The score places more importance on recall than precision. When multiple model summaries are provided, TESLA matches the candidate BNG with each of the model BNGs. The maximum score is taken as the combined score.

### 3.2 TESLA-S: TESLA for Summarization

We adapted TESLA for the nuances of summarization. Mimicking ROUGE-SU4, we construct one matching problem between the unigrams and one between skip bigrams with a window size of four. The two F scores are averaged to give the final score.

The similarity score  $s(x_i, y_j)$  is 1 if the word surface forms of  $x_i$  and  $y_j$  are identical, and 0 otherwise. TESLA has a more sophisticated similarity measure that focuses on awarding partial scores for synonyms and parts of speech (POS) matches. However, the majority of current state-of-the-art summarization systems are extraction-based systems, which do not generate new words. Although our simplistic similarity score may be problematic when evaluating abstract-based systems, the experimental results support our choice of the similarity function. This reflects a major difference between MT and summarization evaluation: while MT systems always generate new sentences, most summarization systems focus on locating existing salient sentences.

Like in TESLA, function words (words in closed POS categories, such as prepositions and articles) have their weights reduced by a factor of 0.1, thus placing more emphasis on the content words. We found this useful empirically.

### 3.3 Significance Test

Koehn (2004) introduced a bootstrap resampling method to compute statistical significance of the difference between two machine translation systems with regard to the BLEU score. We adapt this method to compute the difference between two evaluation metrics in summarization:

1. Randomly choose  $n$  topics from the  $n$  given topics with replacement.
2. Summarize the topics with the list of machine summarizers.
3. Evaluate the list of summaries from Step 2 with the two evaluation metrics under comparison.
4. Determine which metric gives a higher correlation score.
5. Repeat Step 1 – 4 for 1,000 times.

As we have 44 topics in TAC 2011 summarization track,  $n = 44$ . The percentage of times metric  $a$  gives higher correlation than metric  $b$  is said to be the significance level at which  $a$  outperforms  $b$ .

	Initial			Update		
	P	S	K	P	S	K
R-2	0.9606	0.8943	0.7450	0.9029	0.8024	0.6323
R-SU4	<u>0.9806</u>	0.8935	0.7371	0.8847	<u>0.8382</u>	<u>0.6654</u>
BE	0.9388	<u>0.9030</u>	0.7456	<u>0.9057</u>	<u>0.8385</u>	<u>0.6843</u>
4	0.9672	<u>0.9017</u>	0.7351	0.8249	0.8035	0.6070
6	<u>0.9678</u>	0.8816	0.7229	<u>0.9107</u>	<u>0.8370</u>	<u>0.6606</u>
8	0.9555	0.8686	0.7024	0.8981	0.8251	<u>0.6606</u>
10	0.9501	0.8973	<u>0.7550</u>	0.7680	0.7149	0.5504
11	0.9617	0.8937	0.7450	<u>0.9037</u>	0.8018	0.6291
12	<u>0.9739</u>	0.8972	0.7466	0.8559	0.8249	0.6402
13	<u>0.9648</u>	<u>0.9033</u>	<u>0.7582</u>	0.8842	0.7961	0.6276
24	0.9509	0.8997	<u>0.7535</u>	0.8115	0.8199	0.6386
TESLA-S	<b>0.9807</b>	<b>0.9173</b>	<b>0.7734</b>	0.9072	<b>0.8457</b>	0.6811

Table 1: Content correlation with human judgment on summarizer level. Top three scores among AESOP metrics are underlined. The TESLA-S score is bolded when it outperforms all others. ROUGE-2 is shortened to R-2 and ROUGE-SU4 to R-SU4.

### 3.4 Experiments

We test TESLA-S on the AESOP 2011 content evaluation task, judging the metric fitness by comparing its correlations with human judgments for content. The results for the initial and update tasks are reported in Table 1. We show the three baselines (ROUGE-2, ROUGE-SU4, and BE) and submitted metrics with correlations among the top three scores, which are underlined. This setting remains the same for the rest of the experiments. We use three correlation measures: Pearson’s  $r$ , Spearman’s  $\rho$ , and Kendall’s  $\tau$ , represented by P, S, and K, respectively. The ROUGE scores are the recall scores, as per convention. On the initial task, TESLA-S outperforms all metrics on all three correlation measures. On the update task, TESLA-S ranks second, first, and second on Pearson’s  $r$ , Spearman’s  $\rho$ , and Kendall’s  $\tau$ , respectively.

To test how significant the differences are, we perform significance testing using Koehn’s resampling method between TESLA-S and ROUGE-2/ROUGE-SU4, on which TESLA-S is based. The findings are:

- Initial task: TESLA-S is better than ROUGE-2 at 99% significance level as measured by Pearson’s  $r$ .
- Update task: TESLA-S is better than ROUGE-SU4 at 95% significance level as measured by Pearson’s  $r$ .
- All other differences are statistically insignificant, including all correlations on Spearman’s

$\rho$  and Kendall’s  $\tau$ .

The last point can be explained by the fact that Spearman’s  $\rho$  and Kendall’s  $\tau$  are sensitive to only the system rankings, whereas Pearson’s  $r$  is sensitive to the magnitude of the differences as well, hence Pearson’s  $r$  is in general a more sensitive measure.

#### 4 DICOMER: Evaluating Summary Readability

Intuitively, a readable text should also be coherent, and an incoherent text will result in low readability. Both readability and coherence indicate how fluent a text is. We thus hypothesize that a model that measures how coherent a text is can also measure its readability. Lin et al. (2011) introduced discourse role matrix to represent discourse coherence of a text. We first illustrate their model with an example, and then introduce two new feature sources. We then apply the models and evaluate summary readability.

##### 4.1 Lin et al.’s Discourse Coherence Model

First, a free text in Figure 2 is parsed by a discourse parser to derive its discourse relations, which are shown in Figure 3. Lin et al. observed that coherent texts preferentially follow certain relation patterns. However, simply using such patterns to measure the coherence of a text can result in feature sparseness. To solve this problem, they expand the relation sequence into a discourse role matrix, as shown in Table 2. The matrix essentially captures term occurrences in the sentence-to-sentence relation sequences. This model is motivated by the entity-based model (Barzilay and Lapata, 2008) which captures sentence-to-sentence entity transitions. Next, the discourse role transition probabilities of lengths 2 and 3 (e.g., Temp.Arg1→Exp.Arg2 and Comp.Arg1→nil→Temp.Arg1) are calculated with respect to the matrix. For example, the probability of Comp.Arg2→Exp.Arg2 is  $2/25 = 0.08$  in Table 2.

Lin et al. applied their model on the task of discerning an original text from a permuted ordering of its sentences. They modeled it as a pairwise ranking model (i.e., original vs. permuted), and trained a SVM preference ranking model with discourse role

- $S_1$  Japan normally depends heavily on the Highland Valley and Cananea mines as well as the Bougainville mine in Papua New Guinea.
- $S_2$  Recently, Japan has been buying copper elsewhere.
- $S_{3.1}$  But as Highland Valley and Cananea begin operating,
- $S_{3.2}$  they are expected to resume their roles as Japan’s suppliers.
- $S_{4.1}$  According to Fred Demler, metals economist for Drexel Burnham Lambert, New York,
- $S_{4.2}$  “Highland Valley has already started operating
- $S_{4.3}$  and Cananea is expected to do so soon.”

Figure 2: A text with four sentences.  $S_{i,j}$  means the  $j$ th clause in the  $i$ th sentence.

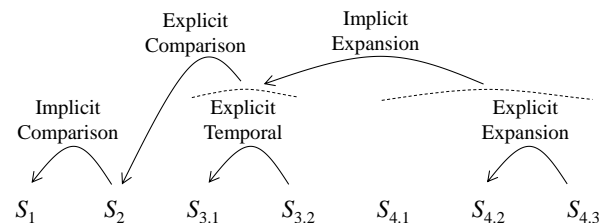


Figure 3: The discourse relations for Figure 2. Arrows are pointing from Arg2 to Arg1.

S#	Terms				
	copper	cananea	operat	depend	...
$S_1$	nil	Comp.Arg1	nil	Comp.Arg1	
$S_2$	Comp.Arg2 Comp.Arg1	nil	nil	nil	
$S_3$	nil	Comp.Arg2 Temp.Arg1 Exp.Arg1	Comp.Arg2 Temp.Arg1 Exp.Arg1	nil	
$S_4$	nil	Exp.Arg2	Exp.Arg1 Exp.Arg2	nil	

Table 2: Discourse role matrix fragment extracted from Figure 2 and 3. Rows correspond to sentences, columns to stemmed terms, and cells contain extracted discourse roles. Temporal, Contingency, Comparison, and Expansion are shortened to Temp, Cont, Comp, and Exp, respectively.

transitions as features and their probabilities as values.

##### 4.2 Two New Feature Sources

We observe that there are two kinds of information in Figure 3 that are not captured by Lin et al.’s

model. The first one is whether a relation is Explicit or Non-Explicit (Lin et al. (2010) termed Non-Explicit to include Implicit, AltLex, EntRel, and NoRel). Explicit relation and Non-Explicit relation have different distributions on each discourse relation (PDTB-Group, 2007). Thus, adding this information may further improve the model. In addition to the set of the discourse roles of “Relation type . Argument tag”, we introduce another set of “Explicit/Non-Explicit . Relation type . Argument tag”. The cell  $C_{cananea,S_3}$  now contains Comp.Arg2, Temp.Arg1, Exp.Arg1, E.Comp.Arg2, E.Temp.Arg1, and N.Exp.Arg1 (E for Explicit and N for Non-Explicit).

The other information that is not in the discourse role matrix is the discourse hierarchy structure, *i.e.*, whether one relation is embedded within another relation. In Figure 3,  $S_{3.1}$  is Arg1 of Explicit Temporal, which is Arg2 of the higher relation Explicit Comparison as well as Arg1 of another higher relation Implicit Expansion. These dependencies are important for us to know how well-structured a summary is. It is represented by the multiple discourse roles in each cell of the matrix. For example, the multiple discourse roles in the cell  $C_{cananea,S_3}$  capture the three dependencies just mentioned. We introduce intra-cell bigrams as a new set of features to the original model: for a cell with multiple discourse roles, we sort them by their surface strings and multiply to obtain the bigrams. For instance,  $C_{cananea,S_3}$  will produce bigrams such as Comp.Arg2 $\leftrightarrow$ Exp.Arg1 and Comp.Arg2 $\leftrightarrow$ Temp.Arg1. When both the Explicit/Non-Explicit feature source and the intra-cell feature source are joined together, it also produces bigram features such as E.Comp.Arg2 $\leftrightarrow$ Temp.Arg1.

### 4.3 Predicting Readability Scores

Lin et al. (2011) used the SVM<sup>light</sup> (Joachims, 1999) package with the preference ranking configuration. To train the model, each source text and one of its permutations form a training pair, where the source text is given a rank of 1 and the permutation is given 0. In testing, the trained model predicts a real number score for each instance, and the instance with the higher score in a pair is said to be the source text.

In the TAC summarization track, human judges scored each model and candidate summary with a readability score from 1 to 5 (5 means most readable). Thus in our setting, instead of a pair of texts, the training input consists of a list of model and candidate summaries from each topic, with their annotated scores as the rankings. Given an unseen test summary, the trained model predicts a real number score. This score essentially is the readability ranking of the test summary. Such ranking can be evaluated by the ranking-based correlations of Spearman’s  $\rho$  and Kendall’s  $\tau$ . As Pearson’s  $r$  measures linear correlation and we do not know whether the real number score follows a linear function, we take the logarithm of this score as the readability score for this instance.

We use the data from AESOP 2009 and 2010 as the training data, and test our metrics on AESOP 2011 data. To obtain the discourse relations of a summary, we use the discourse parser<sup>2</sup> developed in Lin et al. (2010).

### 4.4 Experiments

Table 3 shows the resulting readability correlations. The last four rows show the correlation scores for our coherence model: LIN is the default model by (Lin et al., 2011), LIN+C is LIN with the intra-cell feature class, LIN+E is enhanced with the Explicit/Non-Explicit feature class. We name the LIN model with both new feature sources (*i.e.*, LIN+C+E) DICOMER – a DIScourse COherence Model for Evaluating Readability.

LIN outperforms all metrics on all correlations on both tasks. On the initial task, it outperforms the best scores by 3.62%, 16.20%, and 12.95% on Pearson, Spearman, and Kendall, respectively. Similar gaps (4.27%, 18.52%, and 13.96%) are observed on the update task. The results are much better on Spearman and Kendall. This is because LIN is trained with a ranking model, and both Spearman and Kendall are ranking-based correlations.

Adding either intra-cell or Explicit/Non-Explicit features improves all correlation scores, with Explicit/Non-Explicit giving more pronounced improvements. When both new feature sources are in-

<sup>2</sup><http://wing.comp.nus.edu.sg/~linzihen/parser/>

	Initial			Update		
	P	S	K	P	S	K
R-2	0.7524	0.3975	0.2925	0.6580	0.3732	0.2635
R-SU4	0.7840	0.3953	0.2925	0.6716	0.3627	0.2540
BE	0.7171	0.4091	0.2911	0.5455	0.2445	0.1622
4	<u>0.8194</u>	<u>0.4937</u>	<u>0.3658</u>	<u>0.7423</u>	<u>0.4819</u>	<u>0.3612</u>
6	0.7840	0.4070	0.3036	<u>0.6830</u>	<u>0.4263</u>	<u>0.3141</u>
12	<u>0.7944</u>	<u>0.4973</u>	<u>0.3589</u>	0.6443	0.3991	<u>0.3062</u>
18	<u>0.7914</u>	<u>0.4746</u>	<u>0.3510</u>	0.6698	0.3941	0.2856
23	0.7677	0.4341	0.3162	0.7054	0.4223	0.3014
LIN	<b>0.8556</b>	<b>0.6593</b>	<b>0.4953</b>	<b>0.7850</b>	<b>0.6671</b>	<b>0.5008</b>
LIN+C	<b>0.8612</b>	<b>0.6703</b>	<b>0.4984</b>	<b>0.7879</b>	<b>0.6828</b>	<b>0.5135</b>
LIN+E	<b>0.8619</b>	<b>0.6855</b>	<b>0.5079</b>	<b>0.7928</b>	<b>0.6990</b>	<b>0.5309</b>
DICOMER	<b>0.8666</b>	<b>0.7122</b>	<b>0.5348</b>	<b>0.8100</b>	<b>0.7145</b>	<b>0.5435</b>

Table 3: Readability correlation with human judgment on summarizer level. Top three scores among AESOP metrics are underlined. Our score is bolded when it outperforms all AESOP metrics.

	vs.	Initial			Update		
		P	S	K	P	S	K
LIN		*	**	**	**	**	**
LIN+C	4	**	**	**	**	**	**
LIN+E		**	**	**	*	**	**
DICOMER		**	**	**	**	**	**
DICOMER	LIN	-	*	*	*	-	-

Table 4: Koehn’s significance test for readability. \*\*, \*, and - indicate significance level  $\geq 99\%$ ,  $\geq 95\%$ , and  $< 95\%$ , respectively.

incorporated into the metric, we obtain the best results for all correlation scores: DICOMER outperforms LIN by 1.10%, 5.29%, and 3.95% on the initial task, and 2.50%, 4.74%, and 4.27% on the update task.

Table 3 shows that summarization evaluation Metric 4 tops all other AESOP metrics, except in the case of Spearman’s  $\rho$  on the initial task. We compare our four models to this metric. The results of Koehn’s significance test are reported in Table 4, which demonstrates that all four models outperform Metric 4 significantly. In the last row, we see that when comparing DICOMER to LIN, DICOMER is significantly better on three correlation measures.

## 5 CREMER: Evaluating Overall Responsiveness

With TESLA-S measuring content coverage and DICOMER measuring readability, it is feasible to combine them to predict the overall responsiveness of a summary. There exist many ways to combine two variables mathematically: we can combine them in a linear function or polynomial function, or in a way

	Initial			Update		
	P	S	K	P	S	K
R-2	0.9416	0.7897	0.6096	0.9169	0.8401	0.6778
R-SU4	<u>0.9545</u>	0.7902	0.6017	0.9123	<u>0.8758</u>	<u>0.7065</u>
BE	<u>0.9155</u>	0.7683	0.5673	0.8755	0.7964	0.6254
4	0.9498	<u>0.8372</u>	<u>0.6662</u>	0.8706	<u>0.8674</u>	<u>0.7033</u>
6	<u>0.9512</u>	0.7955	0.6112	<u>0.9271</u>	<u>0.8769</u>	<u>0.7160</u>
11	0.9427	0.7873	0.6064	<u>0.9194</u>	0.8432	0.6794
12	0.9469	<u>0.8450</u>	<u>0.6746</u>	0.8728	0.8611	0.6858
18	0.9480	<u>0.8447</u>	<u>0.6715</u>	0.8912	0.8377	0.6683
23	0.9317	0.7952	0.6080	<u>0.9192</u>	0.8664	0.6953
25	<u>0.9512</u>	0.7899	0.6033	<u>0.9033</u>	0.8139	0.6349
CREMER <sub>LF</sub>	0.9381	0.8346	0.6635	0.8280	0.6860	0.5173
CREMER <sub>PF</sub>	<b>0.9621</b>	<b>0.8567</b>	<b>0.6921</b>	0.8852	0.7863	0.6159
CREMER <sub>RBF</sub>	<b>0.9716</b>	<b>0.8836</b>	<b>0.7206</b>	0.9018	0.8285	0.6588

Table 5: Responsiveness correlation with human judgment on summarizer level. Top three scores among AESOP metrics are underlined. CREMER score is bolded when it outperforms all AESOP metrics.

similar to how precision and recall are combined in  $F$  measure. We applied a machine learning approach to train a regression model for measuring responsiveness. The scores predicted by TESLA-S and DICOMER are used as two features. We use SVM<sup>light</sup> with the regression configuration, testing three kernels: linear function, polynomial function, and radial basis function. We called this model CREMER – a Combined REgression Model for Evaluating Responsiveness.

We train the regression model on AESOP 2009 and 2010 data sets, and test it on AESOP 2011. The DICOMER model that is trained in Section 4 is used to predict the readability scores on all AESOP 2009, 2010, and 2011 summaries. We apply TESLA-S to predict content scores on all AESOP 2009, 2010, and 2011 summaries.

### 5.1 Experiments

The last three rows in Table 5 show the correlation scores of our regression model trained with SVM linear function (LF), polynomial function (PF), and radial basis function (RBF). PF performs better than LF, suggesting that content and readability scores should not be linearly combined. RBF gives better performances than both LF and PF, suggesting that RBF better models the way humans combine content and readability. On the initial task, the model trained with RBF outperforms all submitted metrics. It outperforms the best correlation scores



by 1.71%, 3.86%, and 4.60% on Pearson, Spearman, and Kendall, respectively. All three regression models do not perform as well on the update task. Koehn’s significance test shows that when trained with RBF, CREMER outperforms ROUGE-2 and ROUGE-SU4 on the initial task at a significance level of 99% for all three correlation measures.

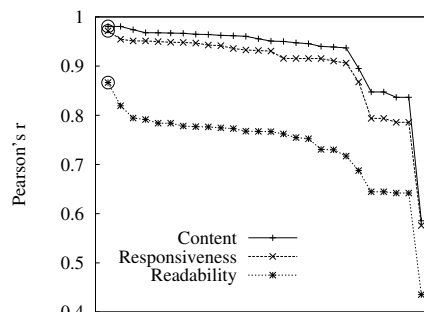
## 6 Discussion

The intuition behind the combined regression model is that combining the readability and content scores will give an overall good responsiveness score. The function to combine them and their weights can be obtained by training. While the results showed that SVM radial basis kernel gave the best performances, this function may not truly mimic how human evaluates responsiveness. Human judges were told to rate summaries by their overall qualities. They may take into account other aspects besides content and readability. Given CREMER did not perform well on the update task, we hypothesize that human judgment of update summaries may involve more complicated rankings or factor in additional input that CREMER currently does not model. We plan to devise a better responsiveness metric in our future work, beyond using a simple combination.

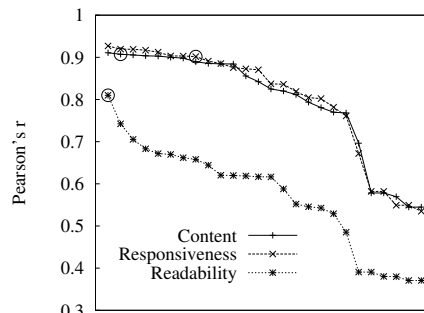
Figure 4 shows a complete picture of Pearson’s  $r$  for all AESOP 2011 metrics and our three metrics on both initial and update tasks. We highlight our metrics with a circle on these curves. On the initial task, correlation scores for content are consistently higher than those for responsiveness with small gaps, whereas on the update task, they are almost overlapping. On the other hand, correlation scores for readability are much lower than those for content and responsiveness, with a gap of about 0.2. Comparing Figure 4a and 4b, evaluation metrics always correlate better on the initial task than on the update task. This suggests that there is much room for improvement for readability metrics, and metrics need to consider update information when evaluating update summarizers.

## 7 Conclusion

We proposed TESLA-S by adapting an MT evaluation metric to measure summary content coverage, and introduced DICOMER by applying a dis-



(a) Evaluation metric values on the initial task.



(b) Evaluation metric values on the update task.

Figure 4: Pearson’s  $r$  for all AESOP 2011 submitted metrics and our proposed metrics. Our metrics are circled. Higher  $r$  value is better.

course coherence model with newly introduced features to evaluate summary readability. We combined these two metrics in the CREMER metric – an SVM-trained regression model – for automatic summarization overall responsiveness evaluation. Experimental results on AESOP 2011 show that DICOMER significantly outperforms all submitted metrics on both initial and update tasks with large gaps, while TESLA-S and CREMER significantly outperform all metrics on the initial task.<sup>3</sup>

## Acknowledgments

This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

<sup>3</sup>Our metrics are publicly available at <http://wing.comp.nus.edu.sg/~linzihen/summeval/>.

## References

- Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34:1–34, March.
- John M. Conroy and Hoa Trang Dang. 2008. Mind the gap: Dangers of divorcing evaluations of summary content from linguistic quality. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, Manchester, UK, August.
- John M. Conroy, Judith D. Schlesinger, Jeff Kubina, Peter A. Rankel, and Dianne P. O’Leary. 2011. CLASSY 2011 at TAC: Guided and multi-lingual summaries and evaluation metrics. In *Proceedings of the Text Analysis Conference 2011 (TAC 2011)*, Gaithersburg, Maryland, USA, November.
- Paulo C. F. de Oliveira. 2011. CatolicaSC at TAC 2011. In *Proceedings of the Text Analysis Conference (TAC 2011)*, Gaithersburg, Maryland, USA, November.
- George Giannakopoulos and Vangelis Karkaletsis. 2011. AutoSummENG and MeMoG in evaluating guided summaries. In *Proceedings of the Text Analysis Conference (TAC 2011)*, Gaithersburg, Maryland, USA, November.
- Eduard Hovy, Chin-Yew Lin, Liang Zhou, and Junichi Fukumoto. 2006. Automated summarization evaluation with basic elements. In *Proceedings of the Fifth Conference on Language Resources and Evaluation (LREC 2006)*.
- Thorsten Joachims. 1999. Making large-scale support vector machine learning practical. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*. MIT Press, Cambridge, MA, USA.
- Alistair Knott. 1996. *A Data-Driven Methodology for Motivating a Set of Coherence Relations*. Ph.D. thesis, Department of Artificial Intelligence, University of Edinburgh.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*.
- Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL 2003)*, Morristown, NJ, USA.
- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2010. A PDTB-styled end-to-end discourse parser. Technical Report TRB8/10, School of Computing, National University of Singapore, August.
- Ziheng Lin, Hwee Tou Ng, and Min-Yen Kan. 2011. Automatically evaluating text coherence using discourse relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, Portland, Oregon, USA, June.
- Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2010. TESLA: Translation evaluation of sentences with linear-programming-based analysis. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, Uppsala, Sweden. Association for Computational Linguistics.
- Ani Nenkova and Rebecca Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proceedings of the 2004 Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL 2004)*, Boston, Massachusetts, USA, May.
- Jun Ping Ng, Praveen Bysani, Ziheng Lin, Min-Yen Kan, and Chew Lim Tan. 2011. SWING: Exploiting category-specific information for guided summarization. In *Proceedings of the Text Analysis Conference 2011 (TAC 2011)*, Gaithersburg, Maryland, USA, November.
- Karolina Owczarzak and Hoa Trang Dang. 2011. Overview of the TAC 2011 summarization track: Guided task and AESOP task. In *Proceedings of the Text Analysis Conference (TAC 2011)*, Gaithersburg, Maryland, USA, November.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, Stroudsburg, PA, USA.
- PDTB-Group, 2007. *The Penn Discourse Treebank 2.0 Annotation Manual*. The PDTB Research Group.
- Emily Pitler, Annie Louis, and Ani Nenkova. 2010. Automatic evaluation of linguistic quality in multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, Stroudsburg, PA, USA.
- Renxian Zhang, You Ouyang, and Wenjie Li. 2011. Guided summarization with aspect recognition. In *Proceedings of the Text Analysis Conference 2011 (TAC 2011)*, Gaithersburg, Maryland, USA, November.
- Liang Zhou, Chin-Yew Lin, Dragos Stefan Munteanu, and Eduard Hovy. 2006. Paraeval: Using paraphrases to evaluate summaries automatically. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2006)*, Stroudsburg, PA, USA.

# Sentence Simplification by Monolingual Machine Translation

**Sander Wubben**  
Tilburg University  
P.O. Box 90135  
5000 LE Tilburg  
The Netherlands  
s.wubben@uvt.nl

**Antal van den Bosch**  
Radboud University Nijmegen  
P.O. Box 9103  
6500 HD Nijmegen  
The Netherlands  
a.vandenbosch@let.ru.nl

**Emiel Krahmer**  
Tilburg University  
P.O. Box 90135  
5000 LE Tilburg  
The Netherlands  
e.j.krahmer@uvt.nl

## Abstract

In this paper we describe a method for simplifying sentences using Phrase Based Machine Translation, augmented with a re-ranking heuristic based on dissimilarity, and trained on a monolingual parallel corpus. We compare our system to a word-substitution baseline and two state-of-the-art systems, all trained and tested on paired sentences from the English part of Wikipedia and Simple Wikipedia. Human test subjects judge the output of the different systems. Analysing the judgements shows that by relatively careful phrase-based paraphrasing our model achieves similar simplification results to state-of-the-art systems, while generating better formed output. We also argue that text readability metrics such as the Flesch-Kincaid grade level should be used with caution when evaluating the output of simplification systems.

## 1 Introduction

Sentence simplification can be defined as the process of producing a simplified version of a sentence by changing some of the lexical material and grammatical structure of that sentence, while still preserving the semantic content of the original sentence, in order to ease its understanding. Particularly language learners (Siddharthan, 2002), people with reading disabilities (Inui et al., 2003) such as aphasia (Carroll et al., 1999), and low-literacy readers (Watanabe et al., 2009) can benefit from this application. It can serve to generate output in a specific limited format, such as subtitles (Daelemans et al., 2004). Sentence simplification can also serve to preprocess the input

of other tasks, such as summarization (Knight and Marcu, 2000), parsing, machine translation (Chandrasekar et al., 1996), semantic role labeling (Vickrey and Koller, 2008) or sentence fusion (Filippova and Strube, 2008).

The goal of simplification is to achieve an improvement in readability, defined as the ease with which a text can be understood. Some of the factors that are known to help increase the readability of text are the vocabulary used, the length of the sentences, the syntactic structures present in the text, and the usage of discourse markers. One effort to create a simple version of English at the vocabulary level has been the creation of Basic English by Charles Kay Ogden. Basic English is a controlled language with a basic vocabulary consisting of 850 words. According to Ogden, 90 percent of all dictionary entries can be paraphrased using these 850 words. An example of a resource that is written using mainly Basic English is the English Simple Wikipedia. Articles on English Simple Wikipedia are similar to articles found in the traditional English Wikipedia, but written using a limited vocabulary (using Basic English where possible). Generally the structure of the sentences in English Simple Wikipedia is less complicated and the sentences are somewhat shorter than those found in English Wikipedia; we offer more detailed statistics below.

### 1.1 Related work

Most earlier work on sentence simplification adopted rule-based approaches. A frequently applied type of rule, aimed to reduce overall sentence length, splits long sentences on the basis of syntactic

information (Chandrasekar and Srinivas, 1997; Carroll et al., 1998; Canning et al., 2000; Vickrey and Koller, 2008). There has also been work on lexical substitution for simplification, where the aim is to substitute difficult words with simpler synonyms, derived from WordNet or dictionaries (Inui et al., 2003).

Zhu et al. (2010) examine the use of paired documents in English Wikipedia and Simple Wikipedia for a data-driven approach to the sentence simplification task. They propose a probabilistic, syntax-based machine translation approach to the problem and compare against a baseline of no simplification and a phrase-based machine translation approach. In a similar vein, Coster and Kauchak (2011) use a parallel corpus of paired documents from Simple Wikipedia and Wikipedia to train a phrase-based machine translation model coupled with a deletion model. Another useful resource is the edit history of Simple Wikipedia, from which simplifications can be learned (Yatskar et al., 2010). Woodsend and Lapata (2011) investigate the use of Simple Wikipedia edit histories and an aligned Wikipedia–Simple Wikipedia corpus to induce a model based on quasi-synchronous grammar. They select the most appropriate simplification by using integer linear programming.

We follow Zhu et al. (2010) and Coster and Kauchak (2011) in proposing that sentence simplification can be approached as a monolingual machine translation task, where the source and target languages are the same and where the output should be simpler in form from the input but similar in meaning. We differ from the approach of Zhu et al. (2010) in the sense that we do not take syntactic information into account; we rely on PBMT to do its work and implicitly learn simplifying paraphrasings of phrases. Our approach differs from Coster and Kauchak (2011) in the sense that instead of focusing on deletion in the PBMT decoding stage, we focus on dissimilarity, as simplification does not necessarily imply shortening (Woodsend and Lapata, 2011), or as the Simple Wikipedia guidelines state, “simpler does not mean short”<sup>1</sup>. Table 1.1 shows the average sentence length and the average

<sup>1</sup>[http://simple.wikipedia.org/wiki/Main\\_Page/Introduction](http://simple.wikipedia.org/wiki/Main_Page/Introduction)

word length for Wikipedia and Simple Wikipedia sentences in the PWKP dataset used in this study (Zhu et al., 2010). These numbers suggest that, although the selection criteria for sentences to be included in this dataset are biased (see Section 2.2), Simple Wikipedia sentences are about 17% shorter, while the average word length is virtually equal.

	Sent. length	Token length
Simple Wikipedia	20.87	4.89
Wikipedia	25.01	5.06

Table 1: Sentence and token length statistics for the PWKP dataset (Zhu et al., 2010).

Statistical machine translation (SMT) has already been successfully applied to the related task of paraphrasing (Quirk et al., 2004; Bannard and Callison-Burch, 2005; Madnani et al., 2007; Callison-Burch, 2008; Zhao et al., 2009; Wubben et al., 2010). SMT typically makes use of large parallel corpora to train a model on. These corpora need to be aligned at the sentence level. Large parallel corpora, such as the multilingual proceedings of the European Parliament (Europarl), are readily available for many languages. Phrase-Based Machine Translation (PBMT) is a form of SMT where the translation model aims to translate longer sequences of words (“phrases”) in one go, solving part of the word ordering problem along the way that would be left to the target language model in a word-based SMT system. PBMT operates purely on statistics and no linguistic knowledge is involved in the process: the phrases that are aligned are motivated statistically, rather than linguistically. This makes PBMT adaptable to any language pair for which there is a parallel corpus available. The PBMT model makes use of a translation model, derived from the parallel corpus, and a language model, derived from a monolingual corpus in the target language. The language model is typically an  $n$ -gram model with smoothing. For any given input sentence, a search is carried out producing an  $n$ -best list of candidate translations, ranked by the decoder score, a complex scoring function including likelihood scores from the translation model, and the target language model. In principle, all of this should be transportable to a data-driven machine translation account of sentence simplification, pro-

vided that a parallel corpus is available that pairs text to simplified versions of that text.

## 1.2 This study

In this work we aim to investigate the use of phrase-based machine translation modified with a dissimilarity component for the task of sentence simplification. While Zhu et al. (2010) have demonstrated that their approach outperforms a PBMT approach in terms of Flesch Reading Ease test scores, we are not aware of any studies that evaluate PBMT for sentence simplification with human judgements. In this study we evaluate the output of Zhu et al. (2010) (henceforth referred to as ‘Zhu’), Woodsend and Lapata (2011) (henceforth referred to as ‘RevILP’), our PBMT based system with dissimilarity-based re-ranking (henceforth referred to as ‘PBMT-R’), a word-substitution baseline, and, as a gold standard, the original Simple Wikipedia sentences. We will first discuss the baseline, followed by the Zhu system, the RevILP system, and our PBMT-R system in Section 2. We then describe the experiment with human judges in Section 3, and its results in Section 4. We close this paper by critically discussing our results in Section 5.

## 2 Sentence Simplification Models

### 2.1 Word-Substitution Baseline

The word substitution baseline replaces words in the source sentence with (near-)synonyms that are more likely according to a language model. For each noun, adjective and verb in the sentence this model takes that word and its part-of-speech tag and retrieves from WordNet all synonyms from all synsets the word occurs in. The word is then replaced by all of its synset words, and each replacement is scored by a SRILM language model (Stolcke, 2002) with probabilities that are obtained from training on the Simple Wikipedia data. The alternative that has the highest probability according to the language model is kept. If no relevant alternative is found, the word is left unchanged. We use the Memory-Based Tagger (Daelemans et al., 1996) trained on the Brown corpus to compute the part-of-speech tags. The WordNet::QueryData<sup>2</sup> Perl mod-

<sup>2</sup><http://search.cpan.org/dist/WordNet-QueryData/QueryData.pm>

ule is used to query WordNet (Fellbaum, 1998).

### 2.2 Zhu et al.

Zhu et al. (2010) learn a sentence simplification model which is able to perform four rewrite operations on the parse trees of the input sentences, namely substitution, reordering, splitting, and deletion. Their model is inspired by syntax-based SMT (Yamada and Knight, 2001) and consists of a language model, a translation model and a decoder. The four mentioned simplification operations together form the translation model. Their model is trained on a corpus containing aligned sentences from English Wikipedia and English Simple Wikipedia called PWKP. The PWKP dataset consists of 108,016 pairs of aligned lines from 65,133 Wikipedia and Simple Wikipedia articles. These articles were paired by following the “interlanguage link”<sup>3</sup>. TF\*IDF at the sentence level was used to align the sentences in the different articles (Nelken and Shieber, 2006).

Zhu et al. (2010) evaluate their system using BLEU and NIST scores, as well as various readability scores that only take into account the output sentence, such as the Flesch Reading Ease test and  $n$ -gram language model perplexity. Although their system outperforms several baselines at the level of these readability metrics, they do not achieve better when evaluated with BLEU or NIST.

### 2.3 RevILP

Woodsend and Lapata’s (2011) model is based on quasi-synchronous grammar (Smith and Eisner, 2006). Quasi-synchronous grammar generates a loose alignment between parse trees. It operates on individual sentences annotated with syntactic information in the form of phrase structure trees. Quasi-synchronous grammar is used to generate all possible rewrite operations, after which integer linear programming is employed to select the most appropriate simplification. Their model is trained on two different datasets: one containing alignments between Wikipedia and English Simple Wikipedia (AlignILP), and one containing alignments between edits in the revision history of Simple Wikipedia (RevILP). RevILP performs best according to the

<sup>3</sup>[http://en.wikipedia.org/wiki/Help:Interlanguage\\_links](http://en.wikipedia.org/wiki/Help:Interlanguage_links)

human judgements conducted in their study. They show that it achieves better scores than Zhu et al. (2010)’s system and is not scored significantly differently from English Simple Wikipedia. In this study we compare against their best performing system, the RevILP system.

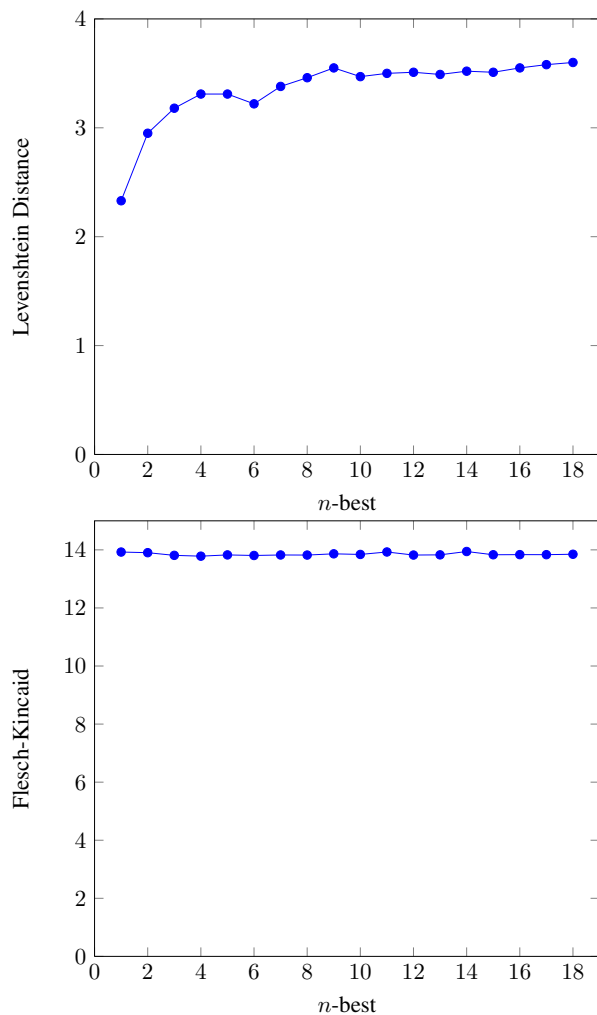


Figure 1: Levenshtein distance and Flesch-Kincaid score of output when varying the  $n$  of the  $n$ -best output of Moses.

## 2.4 PBMT-R

We use the Moses software to train a PBMT model (Koehn et al., 2007). The data we use is the PWKP dataset created by Zhu et al. (2010). In general, a statistical machine translation model finds a best translation  $\tilde{e}$  of a text in language  $f$  to a text in language  $e$  by combining a translation model that

finds the most likely translation  $p(f|e)$  with a language model that outputs the most likely sentence  $p(e)$ :

$$\tilde{e} = \arg \max_{e \in e^*} p(f|e)p(e)$$

The GIZA++ statistical alignment package is used to perform the word alignments, which are later combined into phrase alignments in the Moses pipeline (Och and Ney, 2003) to build the sentence simplification model. GIZA++ utilizes IBM Models 1 to 5 and an HMM word alignment model to find statistically motivated alignments between words. We first tokenize and lowercase all data and use all unique sentences from the Simple Wikipedia part of the PWKP training set to train an  $n$ -gram language model with the SRILM toolkit to learn the probabilities of different  $n$ -grams. Then we invoke the GIZA++ aligner using the training simplification pairs. We run GIZA++ with standard settings and we perform no optimization. This results in a phrase table containing phrase pairs from Wikipedia and Simple Wikipedia and their conditional probabilities as assigned by Moses. Finally, we use the Moses decoder to generate simplifications for the sentences in the test set. For each sentence we let the system generate the ten best distinct solutions (or less, if fewer than ten solutions are generated) as ranked by Moses.

Arguably, dissimilarity is a key factor in simplification (and in paraphrasing in general). As output we would like to be able to select fluent sentences that adequately convey the meaning of the original input, yet that contain differences that operationalize the intended simplification. When training our PBMT system on the PWKP data we may assume that the system learns to simplify automatically, yet there is no aspect of the decoder function in Moses that is sensitive to the fact that it should try to be different from the input – Moses may well translate input to unchanged output, as much of our training data consists of partially equal input and output strings.

To expand the functionality of Moses in the intended direction we perform post-hoc re-ranking on the output based on dissimilarity to the input. We do this to select output that is as different as possible from the source sentence, so that it ideally con-

tains multiple simplifications; at the same time, we base our re-ranking on a top- $n$  of output candidates according to Moses, with a small  $n$ , to ensure that the quality of the output in terms of fluency and adequacy is also controlled for. Setting  $n = 10$ , for each source sentence we re-rank the ten best sentences as scored by the decoder according to the Levenshtein Distance (or edit distance) measure (Levenshtein, 1966) at the word level between the input and output sentence, counting the minimum number of edits needed to transform the source string into the target string, where the allowable edit operations are insertion, deletion, and substitution of a single word. In case of a tie in Levenshtein Distance, we select the sequence with the better decoder score. When Moses is unable to generate ten different sentences, we select from the lower number of outputs. Figure 1 displays Levenshtein Distance and Flesch-Kincaid grade level scores for different values of  $n$ . We use the `Lingua::EN::Fathom` module<sup>4</sup> to calculate Flesch-Kincaid grade level scores. The readability score stays more or less the same, indicating no relation between  $n$  and readability. The average edit distance starts out at just above 2 when selecting the 1-best output string, and increases roughly until  $n = 10$ .

## 2.5 Descriptive statistics

Table 2 displays the average edit distance and the percentage of cases in which no edits were performed for each of the systems and for Simple Wikipedia. We see that the Levenshtein distance between Wikipedia and Simple Wikipedia is the most substantial with an average of 12.3 edits. Given that the average number of tokens is about 25 for Wikipedia and 21 for Simple Wikipedia (cf. Table 1.1), these numbers indicate that the changes in Simple Wikipedia go substantially beyond the average four-word length difference. On average, eight more words are interchanged for other words. About half of the original tokens in the source sentence do not return in the output. Of the three simplification systems, the Zhu system (7.95) and the RevILP (7.18) attain similar edit distances, less substantial than the edits in Simple Wikipedia, but still consid-

<sup>4</sup><http://search.cpan.org/~kimryan/Lingua-EN-Fathom-1.15/lib/Lingua/EN/Fathom.pm>

erable compared to the baseline word-substitution system (4.26) and PBMT-R (3.08). Our system is clearly conservative in its edits.

System	LD	Perc. no edits
Simple Wikipedia	12.30	3
Word Sub	4.26	0
Zhu	7.95	2
RevILP	7.18	22
PBMT-R	3.08	5

Table 2: Levenshtein Distance and percentage of unaltered output sentences.

On the other hand, we observe some differences in the percentage of cases in which the systems decide to produce a sentence identical to the input. In 22 percent of the cases the RevILP system does not alter the sentence. The other systems make this decision about as often as the gold standard, Simple Wikipedia, where only 3% of sentences remain unchanged. The word-substitution baseline always manages to make at least one change.

## 3 Evaluation

### 3.1 Participants

Participants were 46 students of Tilburg University, who participated for partial course credits. All were native speakers of Dutch, and all were proficient in English, having taken a course on Academic English at University level.

### 3.2 Materials

We use the test set used by Zhu et al. (2010) and Woodsend and Lapata (2011). This test set consists of 100 sentences from articles on English Wikipedia, paired with sentences from corresponding articles in English Simple Wikipedia. We selected only those sentences where every system would perform minimally one edit, because we only want to compare the different systems when they actually generate altered, assumedly simplified output. From this subset we randomly pick 20 source sentences, resulting in 20 clusters of one source sentence and 5 simplified sentences, as generated by humans (Simple Wikipedia) and the four systems.

### 3.3 Procedure

The participants were told that they participated in the evaluation of a system that could simplify sentences, and that they would see one source sentence and five automatically simplified versions of that sentence. They were not informed of the fact that we evaluated in fact four different systems and the original Simple Wikipedia sentence. Following earlier evaluation studies (Dodding, 2002; Woodsend and Lapata, 2011), we asked participants to evaluate Simplicity, Fluency and Adequacy of the target headlines on a five point Likert scale. Fluency was defined in the instructions as the extent to which a sentence is proper, grammatical English. Adequacy was defined as the extent to which the sentence has the same meaning as the source sentence. Simplicity was defined as the extent to which the sentence was simpler than the original and thus easier to understand. The order in which the clusters had to be judged was randomized and the order of the output of the various systems was randomized as well.

## 4 Results

### 4.1 Automatic measures

The results of the automatic measures are displayed in Table 3. In terms of the Flesch-Kincaid grade level score, where lower scores are better, the Zhu system scores best, with 7.86 even lower than Simple Wikipedia (8.57). Increasingly worse Flesch-Kincaid scores are produced by RevILP (8.61) and PBMT-R (13.38), while the word substitution baseline scores worst (14.64). With regard to the BLEU score, where Simple Wikipedia is the reference, the PBMT-R system scores highest with 0.43, followed by the RevILP system (0.42) and the Zhu system (0.38). The word substitution baseline scores lowest with a BLEU score of 0.34.

System	Flesch-Kincaid	BLEU
Simple Wikipedia	8.57	1
Word Sub	14.64	0.34
Zhu	7.86	0.38
RevILP	8.61	0.42
PBMT-R	13.38	0.43

Table 3: Flesch-Kincaid grade level and BLEU scores

### 4.2 Human judgements

To test for significance we ran repeated measures analyses of variance with system (Simple Wikipedia, PBMT-R, Zhu, RevILP, word-substitution baseline) as the independent variable, and the three individual metrics as well as their combined mean as the dependent variables. Mauchly’s test for sphericity was used to test for homogeneity of variance, and when this test was significant we applied a Greenhouse-Geisser correction on the degrees of freedom (for the purpose of readability we report the normal degrees of freedom in these cases). Planned pairwise comparisons were made with the Bonferroni method. Table 4 displays these results.

First, we consider the 3 metrics in isolation, beginning with Fluency. We find that participants rated the Fluency of the simplified sentences from the four systems and Simple Wikipedia differently,  $F(4, 180) = 178.436, p < .001, \eta_p^2 = .799$ . The word-substitution baseline, Simple Wikipedia and PBMT-R receive the highest scores (3.86, 3.84 and 3.83 respectively) and don’t achieve significantly different scores on this dimension. All other pairwise comparisons are significant at  $p < .001$ . RevILP attains a score of 3.18, while the Zhu system achieves the lowest mean judgement score of 2.59.

Participants also rated the systems significantly differently on the Adequacy scale,  $F(4, 180) = 116.509, p < .001, \eta_p^2 = .721$ . PBMT-R scores highest (3.71), followed by the word-substitution baseline (3.58), RevILP (3.28), and then by Simple Wikipedia (2.91) and the Zhu system (2.82). Simple Wikipedia and the Zhu system do not differ significantly, and all other pairwise comparisons are significant at  $p < .001$ . The low score of Simple Wikipedia indicates indirectly that the human editors of Simple Wikipedia texts often choose to deviate quite markedly from the meaning of the original text.

Key to the task of simplification are the human judgements of Simplicity. Participants rated the Simplicity of the output from the four systems and Simple Wikipedia differently,  $F(4, 180) = 74.959, p < .001, \eta_p^2 = .625$ . Simple Wikipedia scores highest (3.68) and the word substitution baseline scores lowest (2.42). Between them are the RevILP (2.96), Zhu (2.93) and PBMT-R (2.88) sys-



System	Overall	Fluency	Adequacy	Simplicity
Simple Wikipedia	3.46 (0.39)	3.84 (0.46)	2.91 (0.32)	3.68 (0.39)
Word Sub	3.39 (0.43)	3.86 (0.49)	3.58 (0.35)	2.42 (0.48)
Zhu	2.78 (0.45)	2.59 (0.48)	2.82 (0.37)	2.93 (0.50)
RevILP	3.13 (0.36)	3.18 (0.45)	3.28 (0.32)	2.96 (0.39)
PBMT-R	3.47 (0.46)	3.83 (0.49)	3.71 (0.44)	2.88 (0.46)

Table 4: Mean scores assigned by human subjects, with the standard deviation between brackets

	Adequacy	Simplicity	Flesch-Kincaid	BLEU
Fluency	0.45**	0.24*	0.42**	0.26**
Adequacy		-0.19	0.40**	-0.14
Simplicity			-0.45**	0.42**
Flesch-Kincaid				-0.11

Table 5: Pearson correlation between the different dimensions as assigned by humans and the automatic metrics. Scores marked \* are significant at  $p < .05$  and scores marked \*\* are significant at  $p < .01$

tems, which do not score significantly differently from each other. All other pairwise comparisons are significant at  $p < .001$ .

Finally we report on a combined score created by averaging over the Fluency, Adequacy and Simplicity scores. Inspection of this score, displayed in the leftmost column of Table 4, reveals that the PBMT-R system and Simple Wikipedia score best (3.47 and 3.46 respectively), followed by the word substitution baseline (3.39), which in turn scores higher than RevILP (3.13) and the Zhu system (2.78). We find that participants rated the systems significantly differently overall,  $F(4, 180) = 98.880, p < .001, \eta_p^2 = .687$ . All pairwise comparisons were statistically significant ( $p < .01$ ), except the one between the PBMT-R system and Simple Wikipedia.

### 4.3 Correlations

Table 5 displays the correlations between the scores assigned by humans (Fluency, Adequacy and Simplicity) and the automatic metrics (Flesch-Kincaid and BLEU). We see a significant correlation between Fluency and Adequacy (0.45), as well as between Fluency and Simplicity (0.24). There is a negative significant correlation between Flesch-Kincaid scores and Simplicity (-0.45) while there is a positive significant correlation between Flesch-Kincaid and Adequacy and Fluency. The significant correlations between BLEU and Simplicity (0.42) and Fluency (0.26) are both in the positive direction. There is no significant correlation between BLEU and Ad-

equacy, indicating BLEU’s relative weakness in assessing the semantic overlap between input and output. BLEU and Flesch-Kincaid do not show a significant correlation.

## 5 Discussion

We conclude that a phrase-based machine translation system with added dissimilarity-based re-ranking of the best ten output sentences can successfully be used to perform sentence simplification. Even though the system merely performs phrase-based machine translation and is not specifically geared towards simplification were it not for the dissimilarity-based re-ranking of the output, it performs not significantly differently from state-of-the-art sentence simplification systems in terms of human-judged Simplification. In terms of Fluency and Adequacy our system is judged to perform significantly better. From the relatively low average numbers of edits made by our system we can conclude that our system performs relatively small numbers of changes to the input, that still constitute as sensible simplifications. It does not split sentences (which the Zhu and RevILP systems regularly do); it only rephrases phrases. Yet, it does this better than a word-substitution baseline, which can also be considered a conservative approach; this is reflected in the baseline’s high Fluency score (roughly equal to PBMT-R and Simple Wikipedia) and Adequacy score (only slightly worse than PBMT-R).

Wikipedia	the judge ordered that chapman should receive psychiatric treatment in prison and sentenced him to twenty years to life , slightly less than the maximum possible of twenty-five years to life .
Simple Wikipedia	he was sentenced to twenty-five years to life in prison in 1981 .
Word-substitution baseline	the judge ordered that chapman should have psychiatric treatment in prison and sentenced him to twenty years to life , slightly less than the maximum possible of twenty-five years to life .
Zhu	the judge ordered that chapman should get psychiatric treatment . in prison and sentenced him to twenty years to life , less maximum possible of twenty-five years to life .
RevILP	the judge ordered that chapman should will get psychiatric treatment in prison . he sentenced him to twenty years to life to life .
PBMT-R	the judge ordered that chapman should get psychiatric treatment in prison and sentenced him to twenty years to life , a little bit less than the highest possible to twenty-five years to life .

Table 6: Example output

The output of all systems, the original and the simplified version of an example sentence from the PWKP dataset is displayed in Table 6. The Simple Wikipedia sentences illustrate that significant portions of the original sentences may be dropped, and parts of the semantics of the original sentence discarded. We also see the Zhu and RevILP systems resorting to splitting the original sentence in two, leading to better Flesch-Kincaid scores. The word-substitution baseline changes ‘receive’ in ‘have’, while the PBMT-R system changes the same ‘receive’ in ‘get’, ‘slightly’ to ‘a little bit’, and ‘maximum’ to ‘highest’.

In terms of automatic measures we see that the Zhu system scores particularly well on the Flesch-Kincaid metric, while the RevILP system and our PBMT-R system achieve the highest BLEU scores. We believe that for the evaluation of sentence simplification, BLEU is a more appropriate metric than Flesch-Kincaid or a similar readability metric, although it should be noted that BLEU was found only to correlate significantly with Fluency, not with Adequacy. While BLEU and NIST may be used with this in mind, readability metrics should be avoided altogether in our view. Where machine translation evaluation metrics such as BLEU take into account gold references, readability metrics only take into account characteristics of the sentence such as word length and sentence length, and ignore grammaticality or the semantic adequacy of the content of the output sentence, which BLEU is aimed to implicitly approximate by measuring overlap in  $n$ -grams.

Arguably, readability metrics are best suited to be applied to texts that can be considered grammatical and meaningful, which is not necessarily true for the output of simplification algorithms. A disruptive example that would illustrate this point would be a system that would randomly split original sentences in two or more sequences, achieving considerably lower Flesch-Kincaid scores, yet damaging the grammaticality and semantic coherence of the original text, as is evidenced by the negative correlation for Simplicity and positive correlations for Fluency and Adequacy in Table 5.

In the future we would like to investigate how we can boost the number of edits the system performs, while still producing grammatical and meaning-preserving output. Although the comparison against the Zhu system, which uses syntax-driven machine translation, shows no clear benefit for syntax-based machine translation, it may still be the case that approaches such as Hiero (Chiang et al., 2005) and Joshua (Li et al., 2009), enhanced by dissimilarity-based re-ranking, would improve over our current system. Furthermore, typical simplification operations such as sentence splitting and more radical syntax alterations or even document-level operations such as manipulations of the co-reference structure would be interesting to implement and test

## Acknowledgements

We are grateful to Zhemin Zhu and Kristian Woodsend for sharing their data. We would also like to thank the anonymous reviewers for their comments.

## References

- Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604, Morristown, NJ, USA. Association for Computational Linguistics.
- Chris Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pages 196–205, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Yvonne Canning, John Tait, Jackie Archibald, and Ros Crawley. 2000. Cohesive regeneration of syntactically simplified newspaper text. In *Proceedings of ROMAND 2000*, Lausanne.
- John Carroll, Guido Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. 1998. Practical simplification of English newspaper text to assist aphasic readers. In *AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*, Madison, Wisconsin.
- John Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. 1999. Simplifying text for language-impaired readers. In *Proceedings of EACL'99*, Bergen. ACL.
- R. Chandrasekar and B. Srinivas. 1997. Automatic rules for text simplification. *Knowledge-Based Systems*, 10:183–190.
- Raman Chandrasekar, Christine Doran, and Bangalore Srinivas. 1996. Motivations and methods for text simplification. In *Proceedings of the Sixteenth International Conference on Computational Linguistics (COLING'96)*, pages 1041–1044.
- David Chiang, Adam Lopez, Nitin Madnani, Christof Monz, Philip Resnik, and Michael Subotin. 2005. The hiero machine translation system: extensions, evaluation, and analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05*, pages 779–786, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Will Coster and David Kauchak. 2011. Learning to simplify sentences using wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 1–9, Portland, Oregon, June. Association for Computational Linguistics.
- Walter Daelemans, Jakub Zavrel, Peter Berck, and Steven Gillis. 1996. MBT: A Memory-Based Part of Speech Tagger-Generator. In *Proc. of Fourth Workshop on Very Large Corpora*, pages 14–27. ACL SIGDAT.
- Walter Daelemans, Anja Hothker, and Erik Tjong Kim Sang. 2004. Automatic sentence simplification for subtitling in dutch and english. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 1045–1048.
- George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research, HLT '02*, pages 138–145, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press, May.
- Katja Filippova and Michael Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185, Honolulu, Hawaii, October. Association for Computational Linguistics.
- Kentaro Inui, Atsushi Fujita, Tetsuro Takahashi, Ryu Iida, and Tomoya Iwakura. 2003. Text simplification for reading assistance: A project note. In *Proceedings of the Second International Workshop on Paraphrasing*, pages 9–16, Sapporo, Japan, July. Association for Computational Linguistics.
- Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization – step one: Sentence compression. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI)*, pages 703 – 710, Austin, Texas, USA, July 30 – August 3.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris C. 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*. The Association for Computer Linguistics.
- V. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.
- Zhifei Li, Chris Callison-Burch, Chris Dyer, Juri Ganitkevitch, Sanjeev Khudanpur, Lane Schwartz, Wren N. G. Thornton, Jonathan Weese, and Omar F. Zaidan. 2009. Joshua: an open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 135–139, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Nitin Madnani, Necip Fazil Ayan, Philip Resnik, and Bonnie J. Dorr. 2007. Using paraphrases for parameter tuning in statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation, StatMT '07*, pages 120–127, Stroudsburg, PA, USA. Association for Computational Linguistics.

- Rani Nelken and Stuart M. Shieber. 2006. Towards robust context-sensitive sentence alignment for monolingual corpora. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06)*, Trento, Italy, 3–7 April.
- Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.
- Chris Quirk, Chris Brockett, and William Dolan. 2004. Monolingual machine translation for paraphrase generation. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 142–149, Barcelona, Spain, July. Association for Computational Linguistics.
- Advait Siddharthan. 2002. An architecture for a text simplification system. In *Language Engineering Conference*, page 64. IEEE Computer Society.
- David A. Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the HLT-NAACL Workshop on Statistical Machine Translation*, pages 23–30, New York, June.
- Andreas Stolcke. 2002. SRILM - An Extensible Language Modeling Toolkit. In *In Proc. Int. Conf. on Spoken Language Processing*, pages 901–904, Denver, Colorado.
- D. Vickrey and D. Koller. 2008. Sentence simplification for semantic role labeling. In *Proceedings of the 46th Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Willian Massami Watanabe, Arnaldo Candido Junior, Vincius Rodriguez de Uzlda, Renata Pontin de Mattos Fortes, Thiago Alexandre Salgueiro Pardo, and Sandra M. Alusio. 2009. Facilita: reading assistance for low-literacy readers. In Brad Mehlenbacher, Aristidis Protopsaltis, Ashley Williams, and Shaun Slattery, editors, *SIGDOC*, pages 29–36. ACM.
- Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 409–420, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.
- Sander Wubben, Antal van den Bosch, and Emiel Kraemer. 2010. Paraphrase generation as monolingual translation: data and evaluation. In *Proceedings of the 6th International Natural Language Generation Conference, INLG '10*, pages 203–207, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, ACL '01*, pages 523–530, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Proceedings of the NAACL*, pages 365–368.
- Shiqi Zhao, Xiang Lan, Ting Liu, and Sheng Li. 2009. Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2, ACL '09*, pages 834–842, Stroudsburg, PA, USA. Association for Computational Linguistics.
- Zheming Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 1353–1361, Beijing, China, August. Coling 2010 Organizing Committee.

# A Cost Sensitive Part-of-Speech Tagging: Differentiating Serious Errors from Minor Errors

Hyun-Je Song<sup>1</sup> Jeong-Woo Son<sup>1</sup> Tae-Gil Noh<sup>2</sup> Seong-Bae Park<sup>1,3</sup> Sang-Jo Lee<sup>1</sup>  
<sup>1</sup>School of Computer Sci. & Eng.    <sup>2</sup>Computational Linguistics    <sup>3</sup>NLP Lab.  
Kyungpook Nat'l Univ.    Heidelberg University    Dept. of Computer Science  
Daegu, Korea    Heidelberg, Germany    University of Illinois at Chicago  
{hjsong, jwson, tgnoh}@sejong.knu.ac.kr    sbpark@uic.edu    sjlee@knu.ac.kr

## Abstract

All types of part-of-speech (POS) tagging errors have been equally treated by existing taggers. However, the errors are not equally important, since some errors affect the performance of subsequent natural language processing (NLP) tasks seriously while others do not. This paper aims to minimize these serious errors while retaining the overall performance of POS tagging. Two gradient loss functions are proposed to reflect the different types of errors. They are designed to assign a larger cost to serious errors and a smaller one to minor errors. Through a set of POS tagging experiments, it is shown that the classifier trained with the proposed loss functions reduces serious errors compared to state-of-the-art POS taggers. In addition, the experimental result on text chunking shows that fewer serious errors help to improve the performance of subsequent NLP tasks.

## 1 Introduction

Part-of-speech (POS) tagging is needed as a pre-processor for various natural language processing (NLP) tasks such as parsing, named entity recognition (NER), and text chunking. Since POS tagging is normally performed in the early step of NLP tasks, the errors in POS tagging are critical in that they affect subsequent steps and often lower the overall performance of NLP tasks.

Previous studies on POS tagging have shown high performance with machine learning techniques (Ratnaparkhi, 1996; Brants, 2000; Lafferty et al.,

2001). Among the types of machine learning approaches, supervised machine learning techniques were commonly used in early studies on POS tagging. With the characteristics of a language (Ratnaparkhi, 1996; Kudo et al., 2004) and informative features for POS tagging (Toutanova and Manning, 2000), the state-of-the-art supervised POS tagging achieves over 97% of accuracy (Shen et al., 2007; Manning, 2011). This performance is generally regarded as the maximum performance that can be achieved by supervised machine learning techniques. There have also been many studies on POS tagging with semi-supervised (Subramanya et al., 2010; Sogaard, 2011) or unsupervised machine learning methods (Berg-Kirkpatrick et al., 2010; Das and Petrov, 2011) recently. However, there still exists room to improve supervised POS tagging in terms of error differentiation.

It should be noted that not all errors are equally important in POS tagging. Let us consider the parse trees in Figure 1 as an example. In Figure 1(a), the word “plans” is mistagged as a noun where it should be a verb. This error results in a wrong parse tree that is severely different from the correct tree shown in Figure 1(b). The verb phrase of the verb “plans” in Figure 1(b) is discarded in Figure 1(a) and the whole sentence is analyzed as a single noun phrase. Figure 1(c) and (d) show another tagging error and its effect. In Figure 1(c), a noun is tagged as a NNS (plural noun) where its correct tag is NN (singular or mass noun). However, the error in Figure 1(c) affects only locally the noun phrase to which “physics” belongs. As a result, the general structure of the parse tree in Figure 1(c) is nearly the same as

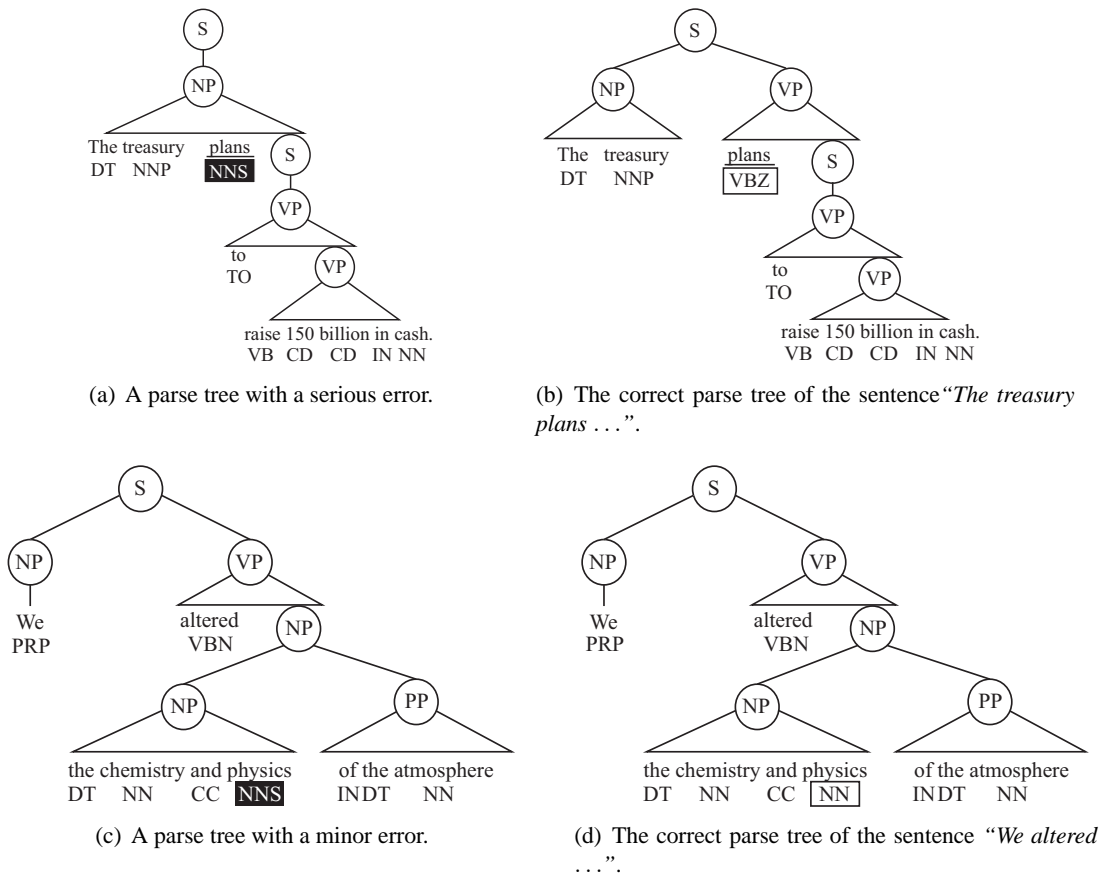


Figure 1: An example of POS tagging errors

the correct one in Figure 1(d). That is, a sentence analyzed with this type of error would yield a correct or near-correct result in many NLP tasks such as machine translation and text chunking.

The goal of this paper is to differentiate the serious POS tagging errors from the minor errors. POS tagging is generally regarded as a classification task, and zero-one loss is commonly used in learning classifiers (Altun et al., 2003). Since zero-one loss considers all errors equally, it can not distinguish error types. Therefore, a new loss is required to incorporate different error types into the learning machines.

This paper proposes two gradient loss functions to reflect differences among POS tagging errors. The functions assign relatively small cost to minor errors, while larger cost is given to serious errors. They are applied to learning multiclass support vector machines (Tsochantaridis et al., 2004) which is trained to minimize the serious errors. Overall accuracy of this SVM is not improved against the state-

of-the-art POS tagger, but the serious errors are significantly reduced with the proposed method. The effect of the fewer serious errors is shown by applying it to the well-known NLP task of text chunking. Experimental results show that the proposed method achieves a higher F1-score compared to other POS taggers.

The rest of the paper is organized as follows. Section 2 reviews the related studies on POS tagging. In Section 3, serious and minor errors are defined, and it is shown that both errors are observable in a general corpus. Section 4 proposes two new loss functions for discriminating the error types in POS tagging. Experimental results are presented in Section 5. Finally, Section 6 draws some conclusions.

## 2 Related Work

The POS tagging problem has generally been solved by machine learning methods for sequential label-

Tag category	POS tags
Substantive	NN, NNS, NNP, NNPS, CD, PRP, PRP\$
Predicate	VB, VBD, VBG, VBN, VBP, VBZ, MD, JJ, JJR, JJS
Adverbial	RB, RBR, RBS, RP, UH, EX, WP, WP\$, WRB, CC, IN, TO
Determiner	DT, PDT, WDT
Etc	FW, SYM, POS, LS

Table 1: Tag categories and POS tags in Penn Tree Bank tag set

ing. In early studies, rich linguistic features and supervised machine learning techniques are applied by using annotated corpora like the Wall Street Journal corpus (Marcus et al., 1994). For instance, Ratnaparkhi (1996) used a maximum entropy model for POS tagging. In this study, the features for rarely appearing words in a corpus are expanded to improve the overall performance. Following this direction, various studies have been proposed to extend informative features for POS tagging (Toutanova and Manning, 2000; Toutanova et al., 2003; Manning, 2011). In addition, various supervised methods such as HMMs and CRFs are widely applied to POS tagging. Lafferty et al. (2001) adopted CRFs to predict POS tags. The methods based on CRFs not only have all the advantages of the maximum entropy markov models but also resolve the well-known problem of label bias. Kudo et al. (2004) modified CRFs for non-segmented languages like Japanese which have the problem of word boundary ambiguity.

As a result of these efforts, the performance of state-of-the-art supervised POS tagging shows over 97% of accuracy (Toutanova et al., 2003; Giménez and Márquez, 2004; Tsuruoka and Tsujii, 2005; Shen et al., 2007; Manning, 2011). Due to the high accuracy of supervised approaches for POS tagging, it has been deemed that there is no room to improve the performance on POS tagging in supervised manner. Thus, recent studies on POS tagging focus on semi-supervised (Spoustová et al., 2009; Subramanya et al., 2010; Søgaard, 2011) or unsupervised approaches (Haghighi and Klein, 2006; Goldwater and Griffiths, 2007; Johnson, 2007; Graca et al., 2009; Berg-Kirkpatrick et al., 2010; Das and Petrov, 2011). Most previous studies on POS tagging have focused on how to extract more linguistic features or how to adopt supervised or unsupervised

approaches based on a single evaluation measure, *accuracy*. However, with a different viewpoint for errors on POS tagging, there is still some room to improve the performance of POS tagging for subsequent NLP tasks, even though the overall accuracy can not be much improved.

In ordinary studies on POS tagging, costs of errors are equally assigned. However, with respect to the performance of NLP tasks relying on the result of POS tagging, errors should be treated differently. In the machine learning community, cost sensitive learning has been studied to differentiate costs among errors. By adopting different misclassification costs for each type of errors, a classifier is optimized to achieve the lowest expected cost (Elkan, 2001; Cai and Hofmann, 2004; Zhou and Liu, 2006).

### 3 Error Analysis of Existing POS Tagger

The effects of POS tagging errors to subsequent NLP tasks vary according to their type. Some errors are serious, while others are not. In this paper, the seriousness of tagging errors is determined by categorical structures of POS tags. Table 1 shows the Penn tree bank POS tags and their categories. There are five categories in this table: *substantive*, *predicate*, *adverbial*, *determiner*, and *etc*. Serious tagging errors are defined as misclassifications among the categories, while minor errors are defined as misclassifications within a category. This definition follows the fact that POS tags in the same category form similar syntax structures in a sentence (Zhao and Marcus, 2009). That is, inter-category errors are treated as serious errors, while intra-category errors are treated as minor errors.

Table 2 shows the distribution of inter-category and intra-category errors observed in section 22–24 of the WSJ corpus (Marcus et al., 1994) that is tagged by the Stanford Log-linear Part-Of-Speech

		Predicted category				
		Substantive	Predicate	Adverbial	Determiner	Etc
True category	Substantive	614	<b>479</b>	<b>32</b>	<b>10</b>	<b>15</b>
	Predicate	<b>585</b>	743	<b>107</b>	<b>2</b>	<b>14</b>
	Adverbial	<b>41</b>	<b>156</b>	500	<b>42</b>	<b>2</b>
	Determiner	<b>13</b>	<b>7</b>	<b>47</b>	24	<b>0</b>
	Etc	<b>23</b>	<b>11</b>	<b>3</b>	<b>1</b>	<b>0</b>

Table 2: The distribution of tagging errors on WSJ corpus by Stanford Part-Of-Speech Tagger.

Tagger (Manning, 2011) (trained with WSJ sections 00–18). In this table, bold numbers denote inter-category errors while all other numbers show intra-category errors. The number of total errors is 3,471 out of 129,654 words. Among them, 1,881 errors (54.19%) are intra-category, while 1,590 of the errors (45.81%) are inter-category. If we can reduce these inter-category errors under the cost of minimally increasing intra-category errors, the tagging results would improve in quality.

Generally in POS tagging, all tagging errors are regarded equally in importance. However, inter-category and intra-category errors should be distinguished. Since a machine learning method is optimized by a loss function, inter-category errors can be efficiently reduced if a loss function is designed to handle both types of errors with different cost. We propose two loss functions for POS tagging and they are applied to multiclass Support Vector Machines.

## 4 Learning SVMs with Class Similarity

POS tagging has been solved as a sequential labeling problem which assumes dependency among words. However, by adopting sequential features such as POS tags of previous words, the dependency can be partially resolved. If it is assumed that words are independent of one another, POS tagging can be regarded as a multiclass classification problem. One of the best solutions for this problem is by using an SVM.

### 4.1 Training SVMs with Loss Function

Assume that a training data set  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_l, y_l)\}$  is given where  $x_i \in \mathbf{R}^d$  is an instance vector and  $y_i \in \{+1, -1\}$  is its class label. SVM finds an optimal hyperplane

satisfying

$$\begin{aligned} x_i \cdot w + b &\geq +1 & \text{for } y_i = +1, \\ x_i \cdot w + b &\leq -1 & \text{for } y_i = -1, \end{aligned}$$

where  $w$  and  $b$  are parameters to be estimated from training data  $D$ . To estimate the parameters, SVMs minimize a hinge loss defined as

$$\begin{aligned} \xi_i &= L_{\text{hinge}}(y_i, w \cdot x_i + b) \\ &= \max\{0, 1 - y_i \cdot (w \cdot x_i + b)\}. \end{aligned}$$

With regularizer  $\|w\|^2$  to control model complexity, the optimization problem of SVMs is defined as

$$\min_{w, \xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i,$$

subject to

$$y_i(x_i \cdot w + b) \geq 1 - \xi_i, \text{ and } \xi_i \geq 0 \quad \forall i,$$

where  $C$  is a user parameter to penalize errors.

Crammer et al. (2002) expanded the binary-class SVM for multiclass classifications. In multiclass SVMs, by considering all classes the optimization of SVM is generalized as

$$\min_{w, \xi} \frac{1}{2} \sum_{k \in K} \|w_k\|^2 + C \sum_{i=1}^l \xi_i,$$

with constraints

$$\begin{aligned} (w_{y_i} \cdot \phi(x_i, y_i)) - (w_k \cdot \phi(x_i, k)) &\geq 1 - \xi_i, \\ \xi_i &\geq 0 \quad \forall i, \quad \forall k \in K \setminus y_i, \end{aligned}$$

where  $\phi(x_i, y_i)$  is a combined feature representation of  $x_i$  and  $y_i$ , and  $K$  is the set of classes.



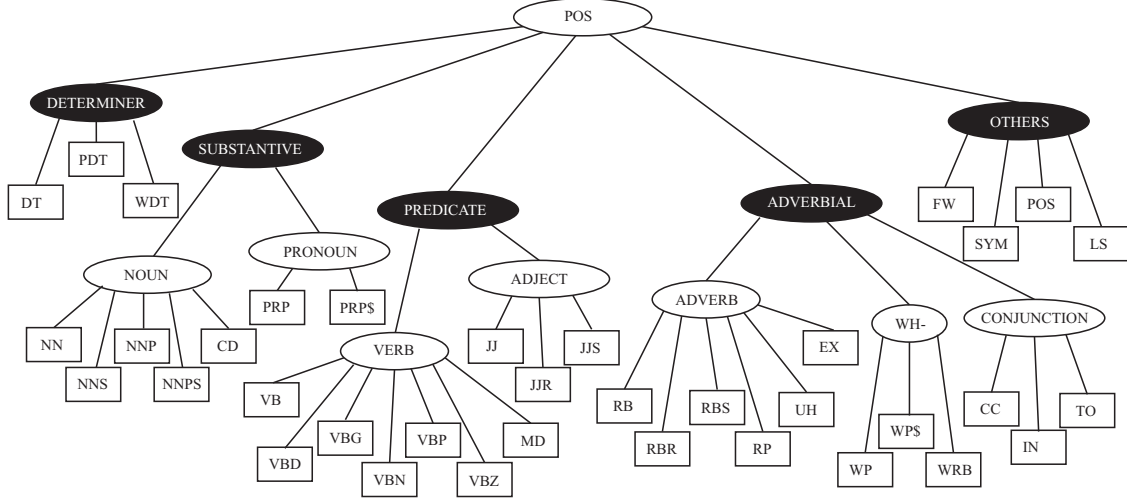


Figure 2: A tree structure of POS tags.

Since both binary and multiclass SVMs adopt a hinge loss, the errors between classes have the same cost. To assign different cost to different errors, Tsochantaridis et al. (2004) proposed an efficient way to adopt arbitrary loss function,  $L(y_i, y_j)$  which returns zero if  $y_i = y_j$ , otherwise  $L(y_i, y_j) > 0$ . Then, the hinge loss  $\xi_i$  is re-scaled with the inverse of the additional loss between two classes. By scaling slack variables with the inverse loss, margin violation with high loss  $L(y_i, y_j)$  is more severely restricted than that with low loss. Thus, the optimization problem with  $L(y_i, y_j)$  is given as

$$\min_{w, \xi} \frac{1}{2} \sum_{k \in K} \|w_k\|^2 + C \sum_{i=1}^l \xi_i, \quad (1)$$

with constraints

$$(w_{y_i} \cdot \phi(x_i, y_i)) - (w_k \cdot \phi(x_i, k)) \geq 1 - \frac{\xi_i}{L(y_i, k)},$$

$$\xi_i \geq 0 \quad \forall i, \quad \forall k \in K \setminus y_i,$$

With the Lagrange multiplier  $\alpha$ , the optimization problem in Equation (1) is easily converted to the following dual quadratic problem.

$$\min_{\alpha} \frac{1}{2} \sum_{i,j} \sum_{k_i \in K \setminus y_i} \sum_{k_j \in K \setminus y_j} \alpha_{i,k_i} \alpha_{j,k_j} \times$$

$$J(x_i, y_i, k_i) J(x_j, y_j, k_j) - \sum_i \sum_{k_i \in K \setminus y_i} \alpha_{i,k_i},$$

with constraints

$$\alpha \geq 0 \quad \text{and} \quad \sum_{k_i \in K \setminus y_i} \frac{\alpha_{i,k_i}}{L(y_i, k_i)} \leq C, \quad \forall i = 1, \dots, l,$$

where  $J(x_i, y_i, k_i)$  is defined as

$$J(x_i, y_i, k_i) = \phi(x_i, y_i) - \phi(x_i, k_i).$$

## 4.2 Loss Functions for POS tagging

To design a loss function for POS tagging, this paper adopts categorical structures of POS tags. The simplest way to reflect the structure of POS tags shown in Table 1 is to assign larger cost to inter-category errors than to intra-category errors. Thus, the loss function with the categorical structure in Table 1 is defined as

$$L_c(y_i, y_j) = \begin{cases} 0 & \text{if } y_i = y_j, \\ \delta & \text{if } y_i \neq y_j \text{ but they belong} \\ & \text{to the same POS category,} \\ 1 & \text{otherwise,} \end{cases} \quad (2)$$

where  $0 < \delta < 1$  is a constant to reduce the value of  $L_c(y_i, y_j)$  when  $y_i$  and  $y_j$  are similar. As shown in this equation, inter-category errors have larger cost than intra-category errors. This loss  $L_c(y_i, y_j)$  is named as *category loss*.

The loss function  $L_c(y_i, y_j)$  is designed to reflect the categories in Table 1. However, the structure of POS tags can be represented as a more complex structure. Let us consider the category, **predicate**.

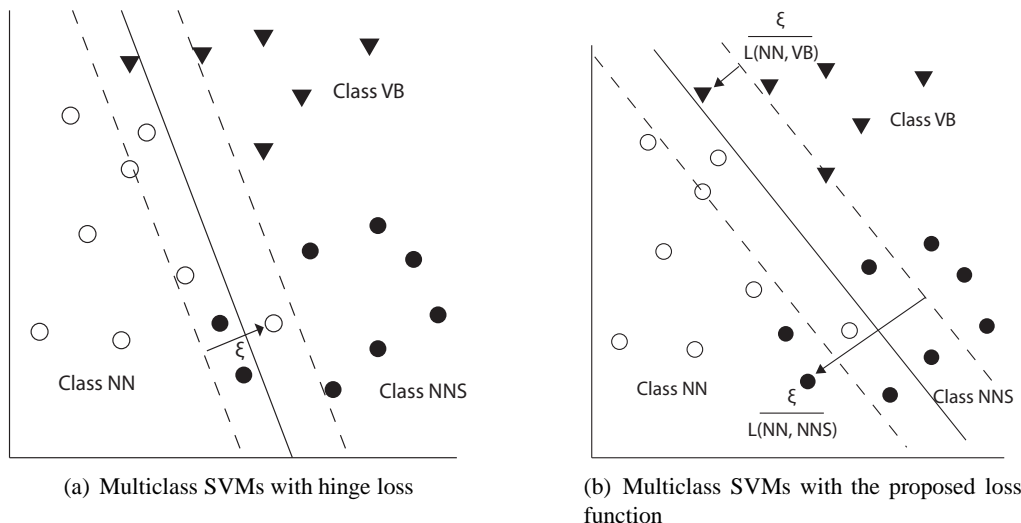


Figure 3: Effect of the proposed loss function in multiclass SVMs

This category has ten POS tags, and can be further categorized into two sub-categories: **verb** and **ad-ject**. Figure 2 represents a categorical structure of POS tags as a tree with five categories of POS tags and their seven sub-categories.

To express the tree structure of Figure 2 as a loss, another loss function  $L_t(y_i, y_j)$  is defined as

$$L_t(y_i, y_j) = \frac{1}{2} [Dist(P_{i,j}, y_i) + Dist(P_{i,j}, y_j)] \times \gamma, \quad (3)$$

where  $P_{i,j}$  denotes the nearest common parent of both  $y_i$  and  $y_j$ , and the function  $Dist(P_{i,j}, y_i)$  returns the number of steps from  $P_{i,j}$  to  $y_i$ . The user parameter  $\gamma$  is a scaling factor of a unit loss for a single step. This loss  $L_t(y_i, y_j)$  returns large value if the distance between  $y_i$  and  $y_j$  is far in the tree structure, and it is named as *tree loss*.

As shown in Equation (1), two proposed loss functions adjust margin violation between classes. They basically assign less value for intra-category errors than inter-category errors. Thus, a classifier is optimized to strictly keep inter-category errors within a smaller boundary. Figure 3 shows a simple example. In this figure, there are three POS tags and two categories. NN (singular or mass noun) and NNS (plural noun) belong to the same category, while VB (verb, base form) is in another category. Figure 3(a) shows the decision boundary of NN based on hinge loss. As shown in this figure, a

single  $\xi$  is applied for the margin violation among all classes. Figure 3(b) also presents the decision boundary of NN, but it is determined with the proposed loss function. In this figure, the margin violation is applied differently to inter-category (NN to VB) and intra-category (NN to NNS) errors. It results in reducing errors between NN and VB even if the errors between NN and NNS could be slightly increased.

## 5 Experiments

### 5.1 Experimental Setting

Experiments are performed with a well-known standard data set, the Wall Street Journal (WSJ) corpus. The data is divided into training, development and test sets as in (Toutanova et al., 2003; Tsuruoka and Tsujii, 2005; Shen et al., 2007). Table 3 shows some simple statistics of these data sets. As shown in this table, training data contains 38,219 sentences with 912,344 words. In the development data set, there are 5,527 sentences with about 131,768 words, those in the test set are 5,462 sentences and 129,654 words. The development data set is used only to select  $\delta$  in Equation (2) and  $\gamma$  in Equation (3).

Table 4 shows the feature set for our experiments. In this table,  $w_i$  and  $t_i$  denote the lexicon and POS tag for the  $i$ -th word in a sentence respectively. We use almost the same feature set as used in (Tsuruoka and Tsujii, 2005) including word features, tag fea-

	Training	Develop	Test
Section	0–18	19–21	22–24
# of sentences	38,219	5,527	5,462
# of terms	912,344	131,768	129,654

Table 3: Simple statistics of experimental data

Feature Name	Description
Word features	$w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2}$ $w_{i-1} \cdot w_i, w_i \cdot w_{i+1}$
Tag features	$t_{i-2}, t_{i-1}, t_{i+1}, t_{i+2}$ $t_{i-2} \cdot t_{i-1}, t_{i+1} \cdot t_{i+2}$ $t_{i-2} \cdot t_{i-1} \cdot t_{i+1}, t_{i-1} \cdot t_{i+1} \cdot t_{i+2}$ $t_{i-2} \cdot t_{i-1} \cdot t_{i+1} \cdot t_{i+2}$
Tag/Word combination	$t_{i-2} \cdot w_i, t_{i-1} \cdot w_i, t_{i+1} \cdot w_i, t_{i+2} \cdot w_i$ $t_{i-1} \cdot t_{i+1} \cdot w_i$
Prefix features	prefixes of $w_i$ (up to length 9)
Suffix features	suffixes of $w_i$ (up to length 9)
Lexical features	whether $w_i$ contains capitals whether $w_i$ has a number whether $w_i$ has a hyphen whether $w_i$ is all capital whether $w_i$ starts with capital and locates at the middle of sentence

Table 4: Feature template for experiments

tures, word/tag combination features, prefix and suffix features as well as lexical features. The POS tags for words are obtained from a two-pass approach proposed by Nakagawa et al. (2001).

In the experiments, two multiclass SVMs with the proposed loss functions are used. One is CL-MSVM with category loss and the other is TL-MSVM with tree loss. A linear kernel is used for both SVMs.

## 5.2 Experimental Results

CL-MSVM with  $\delta = 0.4$  shows the best overall performance on the development data where its error rate is as low as 2.71%.  $\delta = 0.4$  implies that the cost of intra-category errors is set to 40% of that of inter-category errors. The error rate of TL-MSVM is 2.69% when  $\gamma$  is 0.6.  $\delta = 0.4$  and  $\gamma = 0.6$  are set in the all experiments below.

Table 5 gives the comparison with the previous work and proposed methods on the test data. As can be seen from this table, the best performing algorithms achieve near 2.67% error rate (Shen et al., 2007; Manning, 2011). CL-MSVM and TL-MSVM

	Error (%)	# of Intra error	# of Inter error
(Giménez and Màrquez, 2004)	2.84	1,995 (54.11%)	1,692 (45.89%)
(Tsuruoka and Tsujii, 2005)	2.85	-	-
(Shen et al., 2007)	<b>2.67</b>	<b>1,856 (53.52%)</b>	1,612 (46.48%)
(Manning, 2011)	2.68	1,881 (54.19%)	1,590 (45.81%)
CL-MSVM ( $\delta = 0.4$ )	2.69	1,916 (55.01%)	<b>1,567 (44.99%)</b>
TL-MSVM ( $\gamma = 0.6$ )	2.68	1,904 (54.74%)	<b>1,574 (45.26%)</b>

Table 5: Comparison with the previous works

achieve an error rate of 2.69% and 2.68% respectively. Although overall error rates of CL-MSVM and TL-MSVM are not improved compared to the previous state-of-the-art methods, they show reasonable performance.

For inter-category error, CL-MSVM achieves the best performance. The number of inter-category error is 1,567, which shows 23 errors reduction compared to previous best inter-category result by (Manning, 2011). TL-MSVM also makes 16 less inter-category errors than Manning’s tagger. When compared with Shen’s tagger, both CL-MSVM and TL-MSVM make far less inter-category errors even if their overall performance is slightly lower than that of Shen’s tagger. However, the intra-category error rate of the proposed methods has some slight increases. The purpose of proposed methods is to minimize inter-category errors but preserving overall performance. From these results, it can be found that the proposed methods which are trained with the proposed loss functions do differentiate serious and minor POS tagging errors.

## 5.3 Chunking Experiments

The task of chunking is to identify the non-recursive cores for various types of phrases. In chunking, the POS information is one of the most crucial aspects in identifying chunks. Especially inter-category POS errors seriously affect the performance of chunking because they are more likely to mislead the chunk compared to intra-category errors.

Here, chunking experiments are performed with

POS tagger	Accuracy (%)	Precision	Recall	F1-score
(Shen et al., 2007)	96.08	94.03	93.75	93.89
(Manning, 2011)	96.08	94	93.8	93.9
CL-MSVM ( $\delta = 0.4$ )	<b>96.13</b>	<b>94.1</b>	<b>93.9</b>	<b>94.00</b>
TL-MSVM ( $\gamma = 0.6$ )	96.12	<b>94.1</b>	<b>93.9</b>	<b>94.00</b>

Table 6: The experimental results for chunking

a data set provided for the CoNLL-2000 shared task. The training data contains 8,936 sentences with 211,727 words obtained from sections 15–18 of the WSJ. The test data consists of 2,012 sentences and 47,377 words in section 20 of the WSJ. In order to represent chunks, an IOB model is used, where every word is tagged with a chunk label extended with B (the beginning of a chunk), I (inside a chunk), and O (outside a chunk). First, the POS information in test data are replaced to the result of our POS tagger. Then it is evaluated using trained chunking model. Since CRFs (Conditional Random Fields) has been shown near state-of-the-art performance in text chunking (Fei Sha and Fernando Pereira, 2003; Sun et al., 2008), we use CRF++, an open source CRF implementation by Kudo (2005), with default feature template and parameter settings of the package. For simplicity in the experiments, the values of  $\delta$  in Equation (2) and  $\gamma$  in Equation (3) are set to be 0.4 and 0.6 respectively which are same as the previous section.

Table 6 gives the experimental results of text chunking according to the kinds of POS taggers including two previous works, CL-MSVM, and TL-MSVM. Shen’s tagger and Manning’s tagger show nearly the same performance. They achieve an accuracy of 96.08% and around 93.9 F1-score. On the other hand, CL-MSVM achieves 96.13% accuracy and 94.00 F1-score. The accuracy and F1-score of TL-MSVM are 96.12% and 94.00. Both CL-MSVM and TL-MSVM show slightly better performances than other POS taggers. As shown in Table 5, both CL-MSVM and TL-MSVM achieve lower accuracies than other methods, while their inter-category errors are less than that of other experimental methods. Thus, the improvement of CL-MSVM and TL-MSVM implies that, for the subsequent natural language processing, a POS tagger should consider different cost of tagging errors.

## 6 Conclusion

In this paper, we have shown that supervised POS tagging can be improved by discriminating inter-category errors from intra-category ones. An inter-category error occurs by mislabeling a word with a totally different tag, while an intra-category error is caused by a similar POS tag. Therefore, inter-category errors affect the performances of subsequent NLP tasks far more than intra-category errors. This implies that different costs should be considered in training POS tagger according to error types.

As a solution to this problem, we have proposed two gradient loss functions which reflect different costs for two error types. The cost of an error type is set according to (i) categorical difference or (ii) distance in the tree structure of POS tags. Our POS experiment has shown that if these loss functions are applied to multiclass SVMs, they could significantly reduce inter-category errors. Through the text chunking experiment, it is shown that the multiclass SVMs trained with the proposed loss functions which generate fewer inter-category errors achieve higher performance than existing POS taggers.

We have shown that cost sensitive learning can be applied to POS tagging only with multiclass SVMs. However, the proposed loss functions are general enough to be applied to other existing POS taggers. Most supervised machine learning techniques are optimized on their loss functions. Therefore, the performance of POS taggers based on supervised machine learning techniques can be improved by applying the proposed loss functions to learn their classifiers.

## Acknowledgments

This research was supported by the Converging Research Center Program funded by the Ministry of Education, Science and Technology (2011K000659).

## References

- Yasemin Altun, Mark Johnson, and Thomas Hofmann. 2003. Investigating Loss Functions and Optimization Methods for Discriminative Learning of Label Sequences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 145–152.

- Talyor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless Unsupervised Learning with Features. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*. pp. 582–590.
- Thorsten Brants. 2000. TnT-A Statistical Part-of-Speech Tagger. In *Proceedings of the Sixth Applied Natural Language Processing Conference*. pp. 224–231.
- Lijuan Cai and Thomas Hofmann. 2004. Hierarchical Document Categorization with Support Vector Machines. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*. pp. 78–87.
- Koby Crammer, Yoram Singer. 2002. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. *Journal of Machine Learning Research*, Vol. 2. pp. 265–292.
- Dipanjan Das and Slav Petrov. 2011. Unsupervised Part-of-Speech Tagging with Bilingual Graph-Based Projections. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics*. pp. 600–609.
- Charles Elkan. 2001. The Foundations of Cost-Sensitive Learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*. pp. 973–978.
- Jesús Giménez and Lluís Màrquez. 2004. SVMTool: A general POS tagger generator based on Support Vector Machines. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*. pp. 43–46.
- Sharon Goldwater and Thomas T. Griffiths. 2007. A fully Bayesian Approach to Unsupervised Part-of-Speech Tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. pp. 744–751.
- Joao Graca, Kuzman Ganchev, Ben Taskar, and Fernando Pereira. 2009. Posterior vs Parameter Sparsity in Latent Variable Models. In *Advances in Neural Information Processing Systems 22*. pp. 664–672.
- Aria Haghighi and Dan Klein. 2006. Prototype-driven Learning for Sequence Models. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*. pp. 320–327.
- Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the 2007 Joint Meeting of the Conference on Empirical Methods in Natural Language Processing and the Conference on Computational Natural Language Learning*. pp. 296–305.
- Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Applying Conditional Random Fields to Japanese Morphological Analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 230–237.
- Taku Kudo. 2005. CRF++: Yet another CRF toolkit. <http://crfpp.sourceforge.net>.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*. pp. 282–289.
- Christopher D. Manning. 2011. Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?. In *Proceedings of the 12th International Conference on Intelligent Text Processing and Computational Linguistics*. pp. 171–189.
- Tetsuji Nakagawa, Taku Kudo, and Yuji Matsumoto. 2001. Unknown Word Guessing and Part-of-Speech Tagging Using Support Vector Machines. In *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium*. pp. 325–331.
- Adwait Ratnaparkhi. 1996. A Maximum Entropy Model for Part-Of-Speech Tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 133–142.
- Fei Sha and Fernando Pereira. 2003. Shallow Parsing with Conditional Random Fields. In *Proceedings of the Human Language Technology and North American Chapter of the Association for Computational Linguistics*. pp. 213–220.
- Libin Shen, Giorgio Satta, and Aravind K. Joshi. 2007. Guided Learning for Bidirectional Sequence Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. pp. 760–767.
- Anders Søgaard. 2011. Semisupervised condensed nearest neighbor for part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association of Computational Linguistics*. pp. 48–52.
- Drahomíra “johanka” Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. 2009. Semi-supervised training for the averaged perceptron POS tagger. In *Proceedings of the European Chapter of the Association for Computational Linguistics*. pp. 763–771.
- Amarnag Subramanya, Slav Petrov and Fernando Pereira. 2010. Efficient Graph-Based Semi-Supervised Learning of Structured Tagging Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 167–176.
- Xu Sun, Louis-Philippe Morency, Daisuke Okanohara and Jun'ichi Tsujii. 2008. Modeling Latent-Dynamic in Shallow Parsing: A Latent Conditional Model with Improved Inference. In *Proceedings of the 22nd International Conference on Computational Linguistics*. pp. 841–848.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network.

- In *Proceedings of the Human Language Technology and North American Chapter of the Association for Computational Linguistics*. pp. 252–259.
- Kristina Toutanova and Christopher D. Manning. 2000. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 63–70.
- Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemi Altun. 2004. Support Vector Learning for Interdependent and Structured Output Spaces. In *Proceedings of the 21st International Conference on Machine Learning*. pp. 104–111.
- Yoshimasa Tsuruoka and Jun'ichi Tsujii. 2005. Bidirectional Inference with the Easiest-First Strategy for Tagging Sequence Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 467–474.
- Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, Vol. 19, No.2 . pp. 313–330.
- Qiuye Zhao and Mitch Marcus. 2009. A Simple Unsupervised Learner for POS Disambiguation Rules Given Only a Minimal Lexicon. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pp. 688–697.
- Zhi-Hua Zhou and Xu-Ying Liu 2006. On Multi-Class Cost-Sensitive Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. pp. 567–572.

# A Broad-Coverage Normalization System for Social Media Language

Fei Liu Fuliang Weng Xiao Jiang

Research and Technology Center

Robert Bosch LLC

{fei.liu, fuliang.weng}@us.bosch.com

{fixed-term.xiao.jiang}@us.bosch.com

## Abstract

Social media language contains huge amount and wide variety of nonstandard tokens, created both intentionally and unintentionally by the users. It is of crucial importance to normalize the noisy nonstandard tokens before applying other NLP techniques. A major challenge facing this task is the system coverage, i.e., for *any* user-created nonstandard term, the system should be able to restore the correct word within its top  $n$  output candidates. In this paper, we propose a cognitively-driven normalization system that integrates different *human* perspectives in normalizing the nonstandard tokens, including the enhanced letter transformation, visual priming, and string/phonetic similarity. The system was evaluated on both word- and message-level using four SMS and Twitter data sets. Results show that our system achieves over 90% word-coverage across all data sets (a 10% absolute increase compared to state-of-the-art); the broad word-coverage can also successfully translate into message-level performance gain, yielding 6% absolute increase compared to the best prior approach.

## 1 Introduction

The amount of user generated content has increased drastically in the past few years, driven by the prosperous development of the social media websites such as Twitter, Facebook, and Google+. As of June 2011, Twitter has attracted over 300 million users and produces more than 2 billion tweets per week (Twitter, 2011). In a broader sense, Twitter messages, SMS messages, Facebook updates, chat logs, Emails, etc. can all be considered as “social text”,

which is significantly different from the traditional news text due to the informal writing style and the conversational nature. The social text serves as a very valuable information source for many NLP applications, such as the information extraction (Ritter et al., 2011), retrieval (Subramaniam et al., 2009), summarization (Liu et al., 2011a), sentiment analysis (Celikyilmaz et al., 2010), etc. Yet existing systems often perform poorly in this domain due to the extensive use of the nonstandard tokens, emoticons, incomplete and ungrammatical sentences, etc. It is reported that the Stanford named entity recognizer (NER) experienced a performance drop from 90.8% to 45.8% on tweets (Liu et al., 2011c); the part-of-speech (POS) tagger and dependency parser degraded 12.2% and 20.65% respectively on tweets (Foster et al., 2011). It is therefore of great importance to normalize the social text before applying the standard NLP techniques. Text normalization is also crucial for building robust text-to-speech (TTS) systems, which need to determine the pronunciations for nonstandard words in the social text.

The goal of this work is to automatically convert the noisy nonstandard tokens observed in the social text into standard English words. We aim for a robust text normalization system with “broad coverage”, i.e., for *any* user-created nonstandard token, the system should be able to restore the correct word within its top  $n$  candidates ( $n = 1, 3, 10\dots$ ). This is a very challenging task due to two facts: first, there exists huge amount and a wide variety of nonstandard tokens. (Liu et al., 2011b) found more than 4 million distinct out-of-vocabulary tokens in the Edinburgh Twitter corpus (Petrovic et al., 2010); second, the nonstandard tokens consist

2gether (6326)	togetha (919)	tgthr (250)	togeda (20)
2getha (1266)	together (207)	t0gether (57)	togethaa (10)
2gthr (178)	together (94)	together (49)	2getter (10)
u (3240535)	ya (460963)	yo (252274)	yaa (17015)
yaaa (7740)	yew (7591)	yuo (467)	youz (426)
yooooou (186)	youy (105)	yoiu (128)	yoooouuu (82)

Table 1: Nonstandard tokens and their frequencies in the Edinburgh Twitter corpus. The corresponding standard words are “**together**” and “**you**”, respectively.

of a mixture of both unintentional misspellings and intentionally-created tokens for various reasons<sup>1</sup>, including the needs for speed, ease of typing (Crystal, 2009), sentiment expressing (e.g., “coool” (Brody and Diakopoulos, 2011)), intimacy and social purpose (Thurlow, 2003), etc., making it even harder to decipher the social messages. Table 1 shows some example nonstandard tokens.

Existing spell checkers and normalization systems rely heavily on lexical/phonetic similarity to select the correct candidate words. This may not work well since a good portion of the correct words lie outside the specified similarity threshold (e.g., (tomorrow, “tmrw”)<sup>2</sup>), yet the number of candidates increases dramatically as the system strives to increase the coverage by enlarging the threshold. (Han and Baldwin, 2011) reported an average of 127 candidates per nonstandard token with the correct-word coverage of 84%. The low coverage score also enforces an undesirable performance ceiling for the candidate reranking approaches. Different from previous work, we tackle the text normalization problem from a cognitive-sensitive perspective and investigate the human rationales for normalizing the nonstandard tokens. We argue that there exists a set of letter transformation patterns that humans use to decipher the nonstandard tokens. Moreover, the “visual priming” effect may play an important role in human comprehension of the noisy tokens. “Priming” represents an implicit memory effect. For example, if a person reads a list of words including the word *table*, and is later asked to complete a word starting with *tab-*, it is very likely that he answers *table* since the person is *primed*.

In this paper, we propose a broad-coverage normalization system by integrating three human per-

<sup>1</sup>For this reason, we will use the term “nonstandard tokens” instead of “ill-formed tokens” throughout the paper.

<sup>2</sup>We use the form (standard word, “nonstandard token”) to denote an example nonstandard token and its corresponding standard word.

spectives, including the enhanced letter transformation, visual priming, and the string and phonetic similarity. For an arbitrary nonstandard token, the three subnormalizers each suggest their most confident candidates from a different perspective. The candidates can then be heuristically combined or reranked using a message-level decoding process. We evaluate the system on both word- and message-level using four SMS and Twitter data sets. Results show that our system can achieve over 90% word-coverage with limited number of candidates and the broad word-coverage can be successfully translated into message-level performance gain. In addition, our system requires no human annotations, therefore can be easily adapted to different domains.

## 2 Related work

Text normalization, in its traditional sense, is the first step of a speech synthesis system, where the numbers, dates, acronyms, etc. found in the real-world text were converted into standard dictionary words, so that the system can pronounce them correctly. Spell checking plays an important role in this process. (Church and Gale, 1991; Mays et al., 1991; Brill and Moore, 2000) proposed to use the noisy channel framework to generate a list of corrections for any misspelled word, ranked by the corresponding posterior probabilities. (Sproat et al., 2001) enhanced this framework by calculating the likelihood probability as the chance of a noisy token and its associated tag being generated by a specific word.

With the rapid growth of SMS and social media content, text normalization system has drawn increasing attention in the recent decade, where the focus is on converting the noisy nonstandard tokens in the informal text into standard dictionary words. (Choudhury et al., 2007) modeled each standard English word as a hidden Markov model (HMM) and calculated the probability of observing the noisy-token under each of the HMM models; (Cook and Stevenson, 2009) calculated the sum of the probabilities of a noisy token being generated by a specific word and a word formation process; (Beaufort et al., 2010) employed the weighted finite-state machines (FSMs) and rewriting rules for normalizing French SMS; (Pennell and Liu, 2010) focused on tweets created by handsets and developed a CRF tagger for deletion-based abbreviation. The text normalization problem was also tackled under the machine transla-



tion (MT) or speech recognition (ASR) framework. (Aw et al., 2006) adapted a phrase-based MT model for normalizing SMS and achieved satisfying performance. (Kobus et al., 2008) showed that using a statistical MT system in combination with an analogy of the ASR system improved performance in French SMS normalization. (Pennell and Liu, 2011) proposed a two-phase character-level MT system for expanding the abbreviations into standard text.

Recent work also focuses on normalizing the Twitter messages, which is generally considered a more challenging task. (Han and Baldwin, 2011) developed classifiers for detecting the ill-formed words and generated corrections based on the morpho-phonemic similarity. (Liu et al., 2011b) proposed to normalize the nonstandard tokens without explicitly categorizing them. (Xue et al., 2011) adopted the noisy-channel framework and incorporated orthographic, phonetic, contextual, and acronym expansion factors in calculating the likelihood probabilities. (Gouws et al., 2011) revealed that different populations exhibit different shortening styles.

Most of the above systems limit their processing scope to certain categories (e.g., deletion-based abbreviations, misspellings) or require large-scale human annotated corpus for training, which greatly hinders the scalability of the system. In this paper, we propose a novel cognitively-driven text normalization system that robustly tackle both the unintentional misspellings and the intentionally-created noisy tokens. We propose a global context-based approach to purify the automatically collected training data and learn the letter transformation patterns without human supervision. We also propose a cognitively-grounded “visual priming” approach that leverages the “priming” effect to suggest the candidate words. By integrating different perspectives, our system can successfully mimic the human rationales and yield broad word-coverage on both SMS and Twitter messages. To the best of our knowledge, we are the first to integrate these human perspectives in the text normalization system.

### 3 Broad-Coverage Normalization System

In this section, we describe our broad-coverage normalization system, which consists of four key components. For a standard/nonstandard token, three subnormalizers each suggest their most confident



Figure 1: Examples of nonstandard tokens generated by performing letter transformation on the dictionary words.

candidates from a different perspective<sup>3</sup>: “Enhanced Letter Transformation” automatically learns a set of letter transformation patterns and is most effective in normalizing the intentionally created nonstandard tokens through letter insertion, repetition, deletion, and substitution (Section 3.1); “Visual Priming” proposes candidates based on the visual cues and a *primed* perspective (Section 3.2); “Spell Checker” corrects the misspellings (Section 3.3). The fourth component, “Candidate Combination” introduces various strategies to combine the candidates with or without the local context (Section 3.4). Note that it is crucial to integrate different human perspectives so that the system is flexible in processing both unintentional misspellings and various intentionally-created noisy tokens.

#### 3.1 Enhanced Letter Transformation

Given a noisy token  $t_i$  seen in the text, the letter transformation subnormalizer produces a list of correction candidates  $s_i$  under the noisy channel model:

$$\hat{s} = \arg \max_{s_i} p(s_i|t_i) = \arg \max_{s_i} p(t_i|s_i)p(s_i)$$

where we assume each nonstandard token  $t_i$  is dependent on only one English word  $s_i$ , that is, we are not considering acronyms (e.g., “bbl” for “be back later”) in this study.  $p(s_i)$  can be calculated as the unigram count from a background corpus. We formulate the process of generating a nonstandard token  $t_i$  from the dictionary word  $s_i$  using a letter transformation model, and use the model confidence as the probability  $p(t_i|s_i)$ . Figure 1 shows several example (word, token) pairs.

To form a nonstandard token, each letter in the dictionary word can be labeled with: (a) one of the 0-9 digits; (b) one of the 26 characters including itself; (c) the null character “-”; (d) a letter combination<sup>4</sup>. This transformation process from dictio-

<sup>3</sup>For the dictionary word, we allow the subnormalizers to either return the word itself or candidates that are the possibly intended words in the given context (e.g., (with, “wit”).

<sup>4</sup>The set of letter combinations used in this work are {ah, ai, aw, ay, ck, ea, ey, ie, ou, te, wh}

nary words to nonstandard tokens will be learned by a character-level sequence labeling system using the automatically collected (word, token) pairs. Next, we create a large lookup table by applying the character-level labeling system to the standard dictionary words and generate multiple variations for each word using the n-best labeling output, the labeling confidence is used as  $p(t_i|s_i)$ . During testing, we search this lookup table to find the best candidate words for the nonstandard tokens. For tokens with letter repetition, we first generate a set of variants by varying the repetitive letters (e.g.  $C_i = \{\text{“pleas”}, \text{“pleeas”}, \text{“pleaas”}, \text{“pleeaaas”}, \text{“pleeeaaas”}\}$  for  $t_i = \{\text{“pleeeaaas”}\}$ ), then select the maximum posterior probability among all the variants:

$$p(t_i|s_i) = \max_{\tilde{t}_i \in C_i} p(\tilde{t}_i|s_i)$$

Different from the work in (Liu et al., 2011b), we enhanced the letter transformation process with two novel aspects: first, we devise a set of phoneme-, syllable-, morpheme- and word-boundary based features that effectively characterize the formation process of the nonstandard tokens; second, we propose a global context-aware approach to purify the automatically collected training (word, token) pairs, resulting system yielded similar performance but with only one ninth of the original data. We name this subnormalizer “Enhanced Letter Transformation”.

### 3.1.1 Context-Aware Training Pair Selection

Manual annotation of the noisy nonstandard tokens takes a lot of time and effort. (Liu et al., 2011b) proposed to use Google search engine to automatically collect large amount of training pairs. Yet the resulting (work, token) pairs are often noisy, containing pairs such as (events, “ents”), (downtown, “downto”), etc. The ideal training data should consist of the most frequent nonstandard tokens paired with the corresponding corrections, so that the system can learn from the most representative letter transformation patterns.

Motivated by research on word sense disambiguation (WSD) (Mihalcea, 2007), we hypothesize the nonstandard token and the standard word share a lot of common terms in their global context. For example, “luv” and “love” share “i”, “you”, “u”, “it”, etc. among their top context words. Based on this finding, we propose to filter out the low-quality train-

ing pairs by evaluating the global contextual similarity between the word and token. To the best of our knowledge, we are the first to explore this global contextual similarity for the text normalization task.

Given a noisy (word, token) pair, we construct two context vectors  $v_i$  and  $v_j$  by collecting the most frequent terms appearing before or after the work/token. We consider two terms on each side of the word/token as context and restrict the vector length to the top 100 terms. The frequency information were calculated using a large background corpus; stopwords were not excluded from the context vector. The contextual similarity of the (word, token) pair is defined as the cosine similarity between the context vectors  $v_i$  and  $v_j$ :

$$ContextSim(v_i, v_j) = \frac{\sum_{k=1}^n w_{i,k} \times w_{j,k}}{\sqrt{\sum_{k=1}^n w_{i,k}^2} \times \sqrt{\sum_{k=1}^n w_{j,k}^2}}$$

where  $w_{i,k}$  is the weight of term  $t_k$  within the context of term  $t_i$ . The term weights are defined using a normalized TF-IDF method:

$$w_{i,k} = \frac{TF_{i,k}}{TF_i} \times \log\left(\frac{N}{DF_k}\right)$$

where  $TF_{i,k}$  is the count of term  $t_k$  appearing within the context of term  $t_i$ ;  $TF_i$  is the total count of  $t_i$  in the corpus.  $\frac{TF_{i,k}}{TF_i}$  is therefore the relative frequency of  $t_k$  appearing in the context of  $t_i$ ;  $\log\left(\frac{N}{DF_k}\right)$  denotes the inverse document frequency of  $t_k$ , calculated as the logarithm of total tweets ( $N$ ) divided by the number of tweets containing  $t_k$ .

To select the most representative (word, token) pairs for training, we rank the automatically collected 46,288 pairs by the token frequency, filter out pairs whose contextual similarity lower than a threshold  $\theta$  (set empirically at 0.0003), and retain only the top portion (5,000 pairs) for experiments.

### 3.1.2 Character-level Sequence Labeling

For a dictionary word  $s_i$ , we use the conditional random fields (CRF) model to perform character-level labeling to generate its variant  $t_i$ . In the training stage, we align the collected (word, token) pairs at the character level (Liu et al., 2011b), then construct a feature vector for each letter of the dictionary word, using its mapped character as the reference label. This aligned data set is used to train a CRF model (Lafferty et al., 2001; Kudo, 2005)

Character	a	d	v	e	r	t	i	s	e	m	e	n	t	s
Phoneme	AE	D	V	ER	ER	T	AY	Z	_	M	AH	N	T	S
Phoneme boundary	O	O	O	B1	L1	O	O	O	O	O	O	O	O	O
Syllable boundary	B	L	B	I	L	B	I	I	L	B	I	I	I	L
Morpheme boundary	B	I	I	I	I	I	I	L	B	I	I	L	U	
Word boundary	B	I	I	I	I	I	I	I	I	I	I	I	L	

Table 2: Example boundary tags for word “**advertisements**” on the phoneme-, syllable-, morpheme-, and word-level, labeled with the “BILOU” encoding scheme.

with L-BFGS optimization. We use the character/phoneme n-gram and binary vowel features as in (Liu et al., 2011b), but develop a set of boundary features to effectively characterize the letter transformation process.

We notice that in creating the nonstandard tokens, humans tend to drop certain letter units from the word or replace them with other letters. For example, in abbreviating “advertisements” to “ads”, humans may first break the word into smaller units “ad-ver-tise-ment-s”, then drop the middle parts. This also conforms with the word construction theory where a word is composed of smaller units and construction rules. Based on this assumption, we decompose the dictionary words on the phoneme-, syllable-, morpheme-, and word-level<sup>5</sup> and use the “BILOU” tagging scheme (Ratinov and Roth, 2009) to represent the unit boundary, where “BILOU” stands for B(egin), I(nside), L(ast), O(utside), and U(nit-length) of the corresponding unit<sup>6</sup>. Example “BILOU” boundary tags were shown in Table 2.

On top of the boundary tags, we develop a set of conjunction features to accurately pinpoint the current character position. We consider conjunction features formed by concatenating character position in syllable and current syllable position in the word (e.g., conjunction feature “L\_B” for the letter “d” in Table 2). A similar set of features are also developed on morpheme level. We consider conjunction of character/vowel feature and their boundary tags on the syllable/morpheme/word level; conjunction of phoneme and phoneme boundary tags, and absolute position of current character within the corre-

<sup>5</sup>Phoneme decomposition is generated using the (Jiampongamarn et al., 2007) algorithm to map up to two letters to phonemes (2-to-2 alignment); syllable boundary acquired by the hyphenation algorithm (Liang, 1983); morpheme boundary determined by toolkit Morfessor 1.0 (Creutz and Lagus, 2005).

<sup>6</sup>For phoneme boundary, we use “B1” and “L1” to represent two different characters aligned to one phoneme and “B2”, “L2” represent same characters aligned to one phoneme.

sponding syllable/morpheme/word.

We use the aforementioned features to train the CRF model, then apply the model on dictionary words  $s_i$  to generate multiple variations  $t_i$  for each word. When a nonstandard token is seen during testing, we apply the noisy channel to generate a list of best candidate words:  $\hat{s} = \arg \max_{s_i} p(t_i|s_i)p(s_i)$ .

### 3.2 Visual Priming Approach

A second key component of the broad-coverage normalization system is a novel “Visual Priming” subnormalizer. It is built on a cognitively-driven “priming” effect, which has not been explored by other studies yet was shown to be effective across all our data sets.

“Priming”<sup>7</sup> is an implicit memory effect caused by spreading neural networks (Tulving and Stark, 1982). As an example, in the word-stem completion task, participants are given a list of study words, and then asked to complete word “stems” consisting of first 3 letters. A priming effect is observed when participants complete stems with words on the study list more often than with the novel words. The study list activates parts of the human brain right before the stem completion task, later when a word stem is seen, less additional activation is needed for one to choose a word from the study list.

We argue that the “priming” effect may play an important role in human comprehension of the noisy tokens. A person familiarized with the “social talk” is highly *primed* with the most commonly used words; later when a nonstandard token shows only minor visual cues or visual stimulus, it can still be quickly recognized by the person. In this process, the first letter or first few letters of the word serve as a very important visual stimulus. Based on this assumption, we introduce the “priming” subnormalizer based only on the word frequency and the minor visual stimulus. Concretely, this approach proposes candidate words based on the following equation:

$$VisualPrim(s_i|t_i) = \frac{len(LCS(t_i, s_i))}{len(t_i)} \times \log(TF(s_i))$$

Where  $TF(s_i)$  is the term frequency of  $s_i$  as in the background social text corpus;  $\log(TF(s_i))$  *primes* the system with the most common words in the social text;  $LCS(\cdot)$  means the longest common character subsequence;  $len(\cdot)$  denotes the length of the

<sup>7</sup>[http://en.wikipedia.org/wiki/Priming\\_\(psychology\)](http://en.wikipedia.org/wiki/Priming_(psychology))

character sequence. Together  $\frac{\text{len}(LCS(t_i, s_i))}{\text{len}(t_i)}$  provides the minor visual stimulus from  $t_i$ . Note that the first character has been shown to be a crucial visual cue for the brain to understand jumbled words (Davis, ), we therefore consider as candidates only those words  $s_i$  that start with the same character as  $t_i$ . In the case that the nonstandard token  $t_i$  starts with a digit (e.g., “2moro”), we use the mostly likely corresponding letter to search the candidates (those starting with letter “t”). This setting also effectively reduces the candidate search space.

The “visual priming” subnormalizer promotes the candidate words that are frequently used in the social talk and also bear visual similarity with the given noisy token. It slightly deviates from the traditional “priming” notion in that the frequency information were acquired from the global corpus rather than from the prior context. This approach also inherently follows the noisy channel framework, with  $p(t_i|s_i)$  represents the visual stimulus and  $p(s_i)$  being the logarithm of frequency. The candidate words are ranked by  $\hat{s} = \arg \max_{s_i} \text{VisualPrim}(s_i|t_i)$ . We show that the “priming” subnormalizer is robust across data sets abide its simplistic representation.

### 3.3 Spell Checker

The third subnormalizer is the spell checker, which combines the string and phonetic similarity algorithms and is most effective in normalizing the misspellings. We use the Jazzy spell checker (Idzels, 2005) that integrates the DoubleMetaphone phonetic matching algorithm and the Levenshtein distance using the near-miss strategy, which enables the interchange of two adjacent letters, and the replacing/deleting/adding of letters.

### 3.4 Candidate Combination

Each of the three subnormalizers is a stand-alone system and can suggest corrections for the nonstandard tokens. Yet we show that each subnormalizer mimics a different perspective that humans use to decode the nonstandard tokens, as a result, our broad-coverage normalization system is built by integrating candidates from the three subnormalizers using various strategies.

For a noisy token seen in the informal text, the most convenient way of system combination is to harvest up to  $n$  candidates from each of the subnormalizers, and use the pool of candidates (up to

$3n$ ) as the system output. This sets an upper bound for other candidate combination strategies, and we name this approach “**Oracle**”.

A second combination strategy is to give higher priority to candidates from high-precision subsystems. Both “Letter Transformation” and “Spell Checker” have been shown to have high precision in suggesting corrections (Liu et al., 2011b), while “Visual Priming” may not yield high precision due to its definition. We therefore take the top-3 candidates from each of the “Letter Tran.” and “Spell Checker” subsystems, but put candidates from “Letter Tran.” ahead of “Spell Checker” if the confidence of the best candidate is greater than a threshold  $\lambda$  and vice versa. The list of candidates is then compensated using the “Visual Priming” output until the total number reaches  $n$ . We name this approach “**Word-level**” combination since no message-level context information is involved.

Based on the “Word-level” combination output, we can further rerank all the candidates using a message-level Viterbi decoding process (Pennell and Liu, 2011) where the local context information is used to select the best candidate. This approach is named “**Message-level**” combination.

## 4 Experiments

### 4.1 Experimental Setup

We use four SMS and Twitter data sets to evaluate the system effectiveness. Statistics of these data sets are summarized in Table 3. Data set (1) to (3) are used for word-level evaluation; data set (4) for both word- and message-level evaluation. In Table 3, we also present the number of distinct nonstandard tokens found in each data set, and notice that only a small portion of the nonstandard tokens correspond to multiple standard words. We calculate the dictionary coverage of the manually annotated words since this sets an upper bound for any normalization system. We use the Edinburgh Twitter corpus (Petrovic et al., 2010) as the background corpus for frequency calculation, and a dictionary containing 82,324 words.<sup>8</sup> The nonstandard tokens may consist of both numbers/characters and apostrophe.

<sup>8</sup>The dictionary is created by combining the CMU (CMU, 2007) and Aspell (Atkinson, 2006) dictionaries and dropping words with frequency  $< 20$  in the background corpus. “rt” and all single characters except “a” and “i” are excluded.

Index	Domain	Time Period	#Msgs	#Uniq Nonstan. Tokens	%Nonstan. Tkns w/ Multi-cands	%Dict cov. of cands	Reference
(1)	SMS	Around 2007	n/a	303	1.32%	100%	(Choudhury et al., 2007)
(2)	Twitter	Nov 2009 – Feb 2010	6150	3802	3.87%	99.34%	(Liu et al., 2011)
(3)	SMS/Twitter	Aug 2009	4660	2040	2.41%	96.84%	(Pennell and Liu, 2011)
(4)	Twitter	Aug 2010 – Oct 2010	549	558	2.87%	99.10%	(Han and Baldwin, 2011)

Table 3: Statistics of different SMS and Twitter data sets.

The goal of word-level normalization is to convert the list of distinct nonstandard tokens into standard words. For each nonstandard token, the system is considered correct if any of the corresponding standard words is among the  $n$ -best output from the system. We adopt this word-level  $n$ -best accuracy to make our results comparable to other state-of-the-art systems. On message-level, we evaluate the 1-best system output using precision, recall, and  $f$ -score, calculated respective to the nonstandard tokens.

## 4.2 Word-level Results

The word-level results are presented in Table 4, 5, and 6, evaluated on data set (1), (2), (3) respectively. We present the  $n$ -best accuracy ( $n = 1, 3, 10, 20$ ) of the system as well as the ‘‘Oracle’’ results generated by pooling the top-20 candidates from each of the three subnormalizers. The best prior results on the data sets are also included in the tables.

We notice that the broad-coverage system outperforms all other systems on the reported data sets. It achieves about 90% word-level accuracy on data set (1) and (2) with the top-10 candidates (an average 10% performance gain compared to (Liu et al., 2011b)). This is of crucial importance to a normalization system, since the high accuracy and limited number of candidates will enable more sophisticated reranking or supervised learning techniques to select the best candidate. We also observe the ‘‘Oracle’’ system has averagely only 5% gap to the dictionary coverage. A detailed analysis shows that the human annotators perform many semantic/grammar corrections as well as inconsistent annotations, e.g., (sleepy, ‘‘zzz’’), (disliked, ‘‘unliked’’). These are out of the capabilities of the current text normalization system and partly explains the remaining 5% gap.

Regarding the subnormalizer performance, the spell checker yields only 50% to 60% accuracy on all data sets, indicating that the vast amount of the intentionally created nonstandard tokens can hardly be tackled by a system relies solely on the lexical/phonetic similarity. The ‘‘Visual Priming’’ sub-

SMS Dataset (303 pairs)	Word Level Accuracy (%)				
	1-best	3-best	10-best	20-best	Oracle
Jazzy Spell Checker	43.89	55.45	56.77	56.77	n/a
Visual Priming	54.13	74.92	84.82	87.13	n/a
Enhanced Letter Tran.	61.06	74.92	80.86	82.51	n/a
<b>Broad-Cov. System</b>	<b>64.36</b>	<b>80.20</b>	<b>89.77</b>	<b>91.75</b>	<b>94.06</b>
(Pennell et al., 2011)*	60.39	74.58	75.57	75.57	n/a
(Liu et al., 2011)	62.05	75.91	81.19	81.19	n/a
(Cook et al., 2009)	59.4	n/a	83.8	87.8	n/a
(Choudhury et al., 2007)*	59.9	n/a	84.3	88.7	n/a

Table 4: Word-level results on data set (1). \* denotes system requires human annotations for training.

Twitter Dataset (3802 pairs)	Word Level Accuracy (%)				
	1-best	3-best	10-best	20-best	Oracle
Jazzy Spell Checker	47.19	56.92	59.13	59.18	n/a
Visual Priming	54.34	70.59	80.83	84.74	n/a
Enhanced Letter Tran.	61.05	70.07	74.04	74.75	n/a
<b>Broad-Cov. System</b>	<b>69.81</b>	<b>82.51</b>	<b>92.24</b>	<b>93.79</b>	<b>95.71</b>
(Liu et al., 2011)	68.88	78.27	80.93	81.17	n/a

Table 5: Word-level results on data set (2).

normalizer performs surprisingly well and shows robust performance across all data sets. A minor side-effect is that the candidates were restricted to have the same first letter with the noisy token, this sets the upper bound of the approach to 89.77%, 92.45%, and 93.51%, respectively on data set (1), (2), and (3). Compared to other subnormalizers, the ‘‘Enhanced Letter Tran.’’ is effective at normalizing intentionally created tokens and has better precision regarding its top candidate ( $n = 1$ ). We demonstrate the context-aware training pair selection results in Figure 2, by plotting the learning curve using different amounts of training data, ranging from 1,000 (word, token) pairs to the total 46,288 pairs. We notice that the system can effectively learn the letter transformation patterns from a small number of high quality training pairs. The final system was trained using the top 5,000 pairs and the lookup table was created by generating 50 variations for each dictionary word.

## 4.3 Message-level Results

The goal of message-level normalization is to replace each occurrence of the nonstandard token with the candidate word that best fits the local context.

SMS/Twitter Dataset (2404 pairs)	Word Level Accuracy (%)				
	1-best	3-best	10-best	20-best	Oracle
Jazzy Spell Checker	39.89	46.51	48.54	48.67	n/a
Visual Priming	54.12	68.59	78.83	83.11	n/a
Enhanced Letter Tran.	57.65	67.18	71.01	71.88	n/a
<b>Broad-Cov. System</b>	<b>64.39</b>	<b>78.29</b>	<b>86.56</b>	<b>88.69</b>	<b>91.60</b>
(Pennell et al., 2011)*	37.40	n/a	n/a	72.38	n/a

Table 6: Word-level results on data set (3). \* denotes system requires human annotations for training.

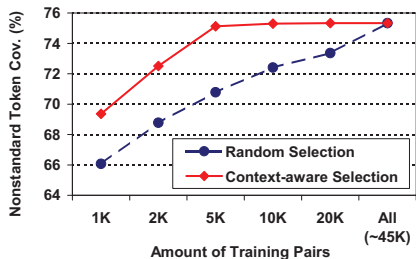


Figure 2: Learning curve of the enhanced letter transformation system using random training pair selection or the context-aware approach. Evaluated on data set (2).

We use the word-level “Broad-Cov. System” for candidate suggestion and the Viterbi algorithm for message-level decoding. The system is evaluated on data set (4) and results shown in Table 7. Following research in (Han and Baldwin, 2011), we focus on the the normalization task and assume perfect non-standard token detection.

The “Word-level w/o Context” results are generated by replacing each nonstandard token using the 1-best word-level candidate. Although the replacement process is static, it results in 70.97% f-score due to the high performance of the word-level system. We explore two language models (LM) for the Viterbi decoding process. First, a bigram LM is trained using the Edinburgh Twitter corpus (53,794,549 English tweets) with the SRILM toolkit (Stolcke, 2002) and Kneser-Ney smoothing; second, we retrieve the bigram probabilities from the Microsoft Web N-gram API (Wang et al., 2010) since this represents a more comprehensive web-based corpus. During decoding, we use the “VisualPrim” score as the emission probability, since this score best fits the log scale and applies to all candidates. For the Twitter LM, we apply a scaling factor of 0.5 to the “VisualPrim” score to make it comparable in scale to the LM probabilities. We use the 3-best word-level candidates for Viterbi decoding. In addition, we add the commonly used corrections for

Twitter Dataset (549 Tweets)		Message-level P/R/F		
		Precision (%)	Recall (%)	F-score (%)
Word-level w/o Context		75.69	66.81	70.97
w/ Context	Web LM	<b>79.12</b>	<b>77.11</b>	<b>78.10</b>
	Twitter LM	<b>84.13</b>	<b>78.38</b>	<b>81.15</b>
(Han and Baldwin, 2011)*		75.30	75.30	75.30

Table 7: Message-level results on data set (4). \* denotes system requires human annotations for training.

16 single-characters, e.g., for “r”, “c”, we add “are”, “see” to the candidate list if they are not already presented. A default “VisualPrim” score ( $\eta = 25$ ) is used for these candidates. As seen from Table 7, both Web LM and Twitter LM achieve better performance than the best prior results, with Twitter LM outperforms the Web LM, yielding a f-score of 81%. This shows that a vanilla Viterbi decoding process is able to outperform the fine-tuned supervised system given competitive word-level candidates. In future, we will investigate other comprehensive message-level candidate reranking process.

## 5 Conclusion

In this paper, we propose a broad-coverage normalization system for the social media language without using the human annotations. It integrates three key components: the enhanced letter transformation, visual priming, and string/phonetic similarity. The system was evaluated on both word- and message-level using four SMS and Twitter data sets. We show that our system achieves over 90% word-coverage across all data sets and the broad word-coverage can be successfully translated into message-level performance gain. We observe that the social media is an emotion-rich language, therefore future normalization system will need to address various sentiment-related expressions, such as emoticons (“:d”, “X-8”), interjections (“bwahaha”, “brrrr”), acronyms (“lol”, “lmao”), etc., whether and how these expressions should be normalized is an unaddressed issue and worths future investigation.

## Acknowledgments

We thank the three anonymous reviewers for their insightful comments and valuable input. We thank Prof. Yang Liu, Deana Pennell, Bo Han, and Prof. Tim Baldwin for sharing the annotated data and the useful discussions. Part of this work was done while Xiao Jiang was a research intern in Bosch Research.

## References

- Kevin Atkinson. 2006. Gnu aspell. <http://aspell.net/>.
- AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for sms text normalization. In *Proceedings of COLING/ACL*, pages 33–40.
- Richard Beaufort, Sophie Roekhaut, Louise-Amélie Cougnon, and Cédric Fairon. 2010. A hybrid rule/model-based finite-state framework for normalizing sms messages. In *Proceedings of ACL*, pages 770–779.
- Eric Brill and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of ACL*.
- Samuel Brody and Nicholas Diakopoulos. 2011. Cooooooooooooooooo!!!!!!!!!!!!!! Using word lengthening to detect sentiment in microblogs. In *Proceedings of EMNLP*, pages 562–570.
- Asli Celikyilmaz, Dilek Hakkani-Tur, and Junlan Feng. 2010. Probabilistic model-based sentiment analysis of twitter messages. In *Proceedings of the IEEE Workshop on Spoken Language Technology*, pages 79–84.
- Monojit Choudhury, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal on Document Analysis and Recognition*, 10(3):157–174.
- Kenneth W. Church and William A. Gale. 1991. Probability scoring for spelling correction. *Statistics and Computing*, 1:93–103.
- CMU. 2007. The cmu pronouncing dictionary. <http://www.speech.cs.cmu.edu/cgi-bin/cmudict>.
- Paul Cook and Suzanne Stevenson. 2009. An unsupervised model for text messages normalization. In *Proceedings of the NAACL HLT Workshop on Computational Approaches to Linguistic Creativity*, pages 71–78.
- Mathias Creutz and Krista Lagus. 2005. Unsupervised morpheme segmentation and morphology induction from text corpora using morfessor 1.0. In *Computer and Information Science, Report A81, Helsinki University of Technology*.
- David Crystal. 2009. Txtng: The gr8 db8. *Oxford University Press*.
- Matt Davis. Reading jumbled texts. <http://www.mrc-cbu.cam.ac.uk/personal/matt.davis/Cmabrigde/>.
- Jennifer Foster, Ozlem Cetinoglu, Joachim Wagner, Joseph Le Roux, Stephen Hogan, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. 2011. #hardtoparse: POS tagging and parsing the twitterverse. In *Proceedings of the AAAI Workshop on Analyzing Microtext*, pages 20–25.
- Stephan Gouws, Donald Metzler, Congxing Cai, and Edward Hovy. 2011. Contextual bearing on linguistic variation in social media. In *Proceedings of the ACL Workshop on Language in Social Media*, pages 20–29.
- Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Mkn sens a #twitter. In *Proceedings of ACL*, pages 368–378.
- Mindaugas Idzelis. 2005. Jazzy: The java open source spell checker. <http://jazzy.sourceforge.net/>.
- Sittichai Jiampojarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden markov models to letter-to-phoneme conversion. In *Proceedings of HLT/NAACL*, pages 372–379.
- Catherine Kobus, François Yvon, and Géraldine Damnati. 2008. Normalizing sms: Are two metaphors better than one? In *Proceedings of COLING*, pages 441–448.
- Taku Kudo. 2005. CRF++: Yet another CRF tool kit. <http://crfpp.sourceforge.net/>.
- John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.
- Franklin Mark Liang. 1983. Word hyphenation by computer. In *PhD Dissertation, Stanford University*.
- Fei Liu, Yang Liu, and Fuliang Weng. 2011a. Why is “sxsw” trending? Exploring multiple text sources for twitter topic summarization. In *Proceedings of the ACL Workshop on Language in Social Media (LSM)*, pages 66–75.
- Fei Liu, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011b. Insertion, deletion, or substitution? Normalizing text messages without pre-categorization nor supervision. In *Proceedings of ACL*, pages 71–76.
- Xiaohua Liu, Shaodan Zhang, Furu Wei, and Ming Zhou. 2011c. Recognizing named entities in tweets. In *Proceedings of ACL*, pages 359–367.
- Eric Mays, Fred J. Damerau, and Robert L. Mercer. 1991. Context based spelling correction. *Information Processing and Management: An International Journal*, 27(5):517–522.
- Rada Mihalcea. 2007. Using wikipedia for automatic word sense disambiguation. In *Proceedings of NAACL*, pages 196–203.
- Deana L. Pennell and Yang Liu. 2010. Normalization of text messages for text-to-speech. In *Proceedings of ICASSP*, pages 4842–4845.
- Deana L. Pennell and Yang Liu. 2011. A character-level machine translation approach for normalization of sms abbreviations. In *Proceedings of the 5th International Joint Conference on Natural Language Processing*, pages 974–982.
- Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2010. The edinburgh twitter corpus. In *Proceedings*

- of the *NAACL HLT Workshop on Computational Linguistics in a World of Social Media*, pages 25–26.
- Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL*, pages 147–155.
- Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of EMNLP*.
- Richard Sproat, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech and Language*, 15(3):287–333.
- Andreas Stolcke. 2002. SRILM – An extensible language modeling toolkit. In *Proceedings of ICSLP*, pages 901–904.
- L. Venkata Subramaniam, Shourya Roy, Tanveer A. Faruque, and Sumit Negi. 2009. A survey of types of text noise and techniques to handle noisy text. In *Proceedings of AND*.
- Crispin Thurlow. 2003. Generation txt? the sociolinguistics of young people’s text-messaging. *Discourse Analysis Online*.
- Endel Tulving and Daniel L. Schacter; Heather A. Stark. 1982. Priming effects in word fragment completion are independent of recognition memory. *Journal of Experimental Psychology: Learning, Memory and Cognition*, 8(4).
- Twitter. 2011. <http://en.wikipedia.org/wiki/Twitter>.
- Kuansan Wang, Christopher Thrasher, Evelyne Viegas, Xiaolong Li, and Bo june (Paul) Hsu. 2010. An overview of microsoft web n-gram corpus and applications. In *Proceedings of NAACL-HLT*, pages 45–48.
- Zhenzhen Xue, Dawei Yin, and Brian D. Davison. 2011. Normalizing microtext. In *Proceedings of the AAAI Workshop on Analyzing Microtext*, pages 74–79.



# Incremental Joint Approach to Word Segmentation, POS Tagging, and Dependency Parsing in Chinese

Jun Hatori<sup>1</sup> Takuya Matsuzaki<sup>2</sup> Yusuke Miyao<sup>2</sup> Jun'ichi Tsujii<sup>3</sup>

<sup>1</sup>University of Tokyo / 7-3-1 Hongo, Bunkyo, Tokyo, Japan

<sup>2</sup>National Institute of Informatics / 2-1-2 Hitotsubashi, Chiyoda, Tokyo, Japan

<sup>3</sup>Microsoft Research Asia / 5 Danling Street, Haidian District, Beijing, P.R. China

hatori@is.s.u-tokyo.ac.jp

{takuya-matsuzaki, yusuke}@nii.ac.jp jtsujii@microsoft.com

## Abstract

We propose the first joint model for word segmentation, POS tagging, and dependency parsing for Chinese. Based on an extension of the incremental joint model for POS tagging and dependency parsing (Hatori et al., 2011), we propose an efficient character-based decoding method that can combine features from state-of-the-art segmentation, POS tagging, and dependency parsing models. We also describe our method to align comparable states in the beam, and how we can combine features of different characteristics in our incremental framework. In experiments using the Chinese Treebank (CTB), we show that the accuracies of the three tasks can be improved significantly over the baseline models, particularly by 0.6% for POS tagging and 2.4% for dependency parsing. We also perform comparison experiments with the partially joint models.

## 1 Introduction

In processing natural languages that do not include delimiters (e.g. spaces) between words, word segmentation is the crucial first step that is necessary to perform virtually all NLP tasks. Furthermore, the word-level information is often augmented with the POS tags, which, along with segmentation, form the basic foundation of statistical NLP.

Because the tasks of word segmentation and POS tagging have strong interactions, many studies have been devoted to the task of joint word segmentation and POS tagging for languages such as Chinese (e.g. Kruengkrai et al. (2009)). This is because some of the segmentation ambiguities cannot be resolved without considering the surrounding grammatical constructions encoded in a sequence of POS tags. The joint approach to word segmentation and POS tagging has been reported to improve word segmentation and POS tagging accuracies by more than

1% in Chinese (Zhang and Clark, 2008). In addition, some researchers recently proposed a joint approach to Chinese POS tagging and dependency parsing (Li et al., 2011; Hatori et al., 2011); particularly, Hatori et al. (2011) proposed an incremental approach to this joint task, and showed that the joint approach improves the accuracies of these two tasks.

In this context, it is natural to consider further a question regarding the joint framework: how strongly do the tasks of word segmentation and dependency parsing interact? In the following Chinese sentences:

当今 和平奖 与 和平 事业 相关  
current peace-prize and peace operation related  
*The current peace prize and peace operations are related.*

当今 和平 奖与 和平 事业 相关 团体  
current peace award peace operation related group  
*The current peace is awarded to peace-operation-related groups.*

the only difference is the existence of the last word 团体; however, whether or not this word exists changes the whole syntactic structure and segmentation of the sentence. This is an example in which word segmentation cannot be handled properly without considering long-range syntactic information.

Syntactic information is also considered beneficial to improve the segmentation of out-of-vocabulary (OOV) words. Unlike languages such as Japanese that use a distinct character set (i.e. *katakana*) for foreign words, the transliterated words in Chinese, many of which are OOV words, frequently include characters that are also used as common or function words. In the current systems, the existence of these characters causes numerous over-segmentation errors for OOV words.

Based on these observations, we aim at building a joint model that simultaneously processes word segmentation, POS tagging, and dependency parsing, trying to capture global interaction among

these three tasks. To handle the increased computational complexity, we adopt the incremental parsing framework with dynamic programming (Huang and Sagae, 2010), and propose an efficient method of character-based decoding over candidate structures.

Two major challenges exist in formalizing the joint segmentation and dependency parsing task in the character-based incremental framework. First, we must address the problem of how to align comparable states effectively in the beam. Because the number of dependency arcs varies depending on how words are segmented, we devise a step alignment scheme using the number of character-based arcs, which enables effective joint decoding for the three tasks.

Second, although the feature set is fundamentally a combination of those used in previous works (Zhang and Clark, 2010; Huang and Sagae, 2010), to integrate them in a single incremental framework is not straightforward. Because we must perform decisions of three kinds (segmentation, tagging, and parsing) in an incremental framework, we must adjust which features are to be activated when, and how they are combined with which action labels. We have also found that we must balance the learning rate between features for segmentation and tagging decisions, and those for dependency parsing.

We perform experiments using the Chinese Treebank (CTB) corpora, demonstrating that the accuracies of the three tasks can be improved significantly over the pipeline combination of the state-of-the-art joint segmentation and POS tagging model, and the dependency parser. We also perform comparison experiments with partially joint models, and investigate the tradeoff between the running speed and the model performance.

## 2 Related Works

In Chinese, Luo (2003) proposed a joint constituency parser that performs segmentation, POS tagging, and parsing within a single character-based framework. They reported that the POS tags contribute to segmentation accuracies by more than 1%, but the syntactic information has no substantial effect on the segmentation accuracies. In contrast, we built a joint model based on a dependency-based framework, with a rich set of structural features. Using it, we show the first positive result in Chinese that the segmentation accuracies can be improved using the syntactic information.

Another line of work exists on lattice-based parsing for Semitic languages (Cohen and Smith, 2007; Goldberg and Tsarfaty, 2008). These methods first convert an input sentence into a lattice encoding the morphological ambiguities, and then conduct joint morphological segmentation and PCFG parsing. However, the segmentation possibilities considered in those studies are limited to those output by an existing morphological analyzer. In addition, the lattice does not include word segmentation ambiguities crossing boundaries of space-delimited tokens. In contrast, because the Chinese language does not have spaces between words, we fundamentally need to consider the lattice structure of the whole sentence. Therefore, we place no restriction on the segmentation possibilities to consider, and we assess the full potential of the joint segmentation and dependency parsing model.

Among the many recent works on joint segmentation and POS tagging for Chinese, the linear-time incremental models by Zhang and Clark (2008) and Zhang and Clark (2010) largely inspired our model. Zhang and Clark (2008) proposed an incremental joint segmentation and POS tagging model, with an effective feature set for Chinese. However, it requires to computationally expensive multiple beams to compare words of different lengths using beam search. More recently, Zhang and Clark (2010) proposed an efficient character-based decoder for their word-based model. In their new model, a single beam suffices for decoding; hence, they reported that their model is practically ten times as fast as their original model. To incorporate the word-level features into the character-based decoder, the features are decomposed into substring-level features, which are effective for incomplete words to have comparable scores to complete words in the beam. Because we found that even an incremental approach with beam search is intractable if we perform the word-based decoding, we take a character-based approach to produce our joint model.

The incremental framework of our model is based on the joint POS tagging and dependency parsing model for Chinese (Hatori et al., 2011), which is an extension of the shift-reduce dependency parser with dynamic programming (Huang and Sagae, 2010). They specifically modified the shift action so that it assigns the POS tag when a word is shifted onto the stack. However, because they regarded word segmentation as given, their model did not consider the

interaction between segmentation and POS tagging.

### 3 Model

#### 3.1 Incremental Joint Segmentation, POS Tagging, and Dependency Parsing

Based on the joint POS tagging and dependency parsing model by Hatori et al. (2011), we build our joint model to solve word segmentation, POS tagging, and dependency parsing within a single framework. Particularly, we change the role of the shift action and additionally use the append action, inspired by the character-based actions used in the joint segmentation and POS tagging model by Zhang and Clark (2010).

The list of actions used is the following:

- A: *append* the first character in the queue to the word on top of the stack.
- SH( $t$ ): *shift* the first character in the input queue as a new word onto the stack, with POS tag  $t$ .
- RL/RR: *reduce* the top two trees on the stack,  $(s_0, s_1)$ , into a subtree  $s_0 \hat{\ } s_1 / s_0 \hat{\ } s_1$ , respectively.

Although SH( $t$ ) is similar to the one used in Hatori et al. (2011), now it shifts the first character in the queue as a new word, instead of shifting a word. Following Zhang and Clark (2010), the POS tag is assigned to the word when its first character is shifted, and the word–tag pairs observed in the training data and the closed-set tags (Xia, 2000) are used to prune unlikely derivations. Because 33 tags are defined in the CTB tag set (Xia, 2000), our model exploits a total of 36 actions.

To train the model, we use the averaged perceptron with the early update (Collins and Roark, 2004). In our joint model, the early update is invoked by mistakes in any of word segmentation, POS tagging, or dependency parsing.

#### 3.2 Alignment of States

When dependency parsing is integrated into the task of joint word segmentation and POS tagging, it is not straightforward to define a scheme to *align (synchronize)* the states in the beam. In beam search, we use the *step index* that is associated with each state: the parser states in process are aligned according to the index, and the beam search pruning is applied to those states with the same index. Consequently, for the beam search to function effectively, all states with the same index must be *comparable*, and all terminal states should have the same step index.

We can first think of using the number of shifted characters as the step index, as Zhang and Clark (2010) does. However, because RL/RR actions can be performed without incrementing the step index, the decoder tends to prefer states with more dependency arcs, resulting more likely in premature choice of ‘reduce’ actions or oversegmentation of words. Alternatively, we can consider using the number of actions that have been applied as the step index, as Hatori et al. (2011) does. However, this results in inconsistent numbers of actions to reach the terminal states: some states that segment words into larger chunks reach a terminal state earlier than other states with smaller chunks. For these reasons, we have found that both approaches yield poor models that are not at all competitive with the baseline (pipeline) models<sup>1</sup>.

To address this issue, we propose an indexing scheme using the number of character-based arcs. We presume that in addition to the word-to-word dependency arcs, each word (of length  $M$ ) implicitly has  $M - 1$  inter-character arcs, as in:  $\boxed{A \hat{\ } B \hat{\ } C}$ ,  $\boxed{A \hat{\ } B} \hat{\ } \boxed{C}$ , and  $\boxed{A} \hat{\ } \boxed{B} \hat{\ } \boxed{C}$  (each rectangle denotes a word). Then we can define the step index as the sum of the number of shifted characters and the total number of (inter-word and intra-word) dependency arcs, which thereby meets all the following conditions:

- (1) All subtrees spanning  $M$  consecutive characters have the same index  $2M - 1$ .
- (2) All terminal states have the same step index  $2N$  (including the root arc), where  $N$  is the number of characters in the sentence.
- (3) Every action increases the index.

Note that the number of shifted characters is also necessary to meet condition (3). Otherwise, it allows an unlimited number of SH( $t$ ) actions without incrementing the step index. Figure 1 portrays how the states are aligned using the proposed scheme, where a subtree is denoted as a rectangle with its partial index shown inside it.

In our framework, because an action increases the step index by 1 (for SH( $t$ ) or RL/RR) or 2 (for A), we need to use two beams to store new states at each step. The computational complexity of the entire process is  $O(B(T + 3) \cdot 2N)$ , where  $B$  is the beam

<sup>1</sup>For example, in our preliminary experiment on CTB-5, the step indexing according to the number of actions underperforms the baseline model by 0.2–0.3% in segmentation accuracy.

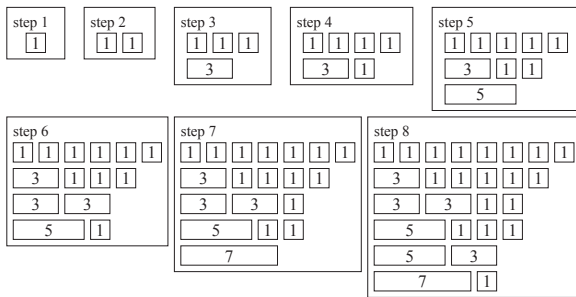


Figure 1: Illustration of the alignment of steps.

size,  $T$  is the number of POS tags ( $= 33$ ), and  $N$  is the number of characters in the sentence. Theoretically, the computational time is greater than that with the character-based joint segmentation and tagging model by Zhang and Clark (2010) by a factor of  $\frac{T+3}{T+1} \cdot \frac{2N}{N} \simeq 2.1$ , when the same beam size is used.

### 3.3 Features

The feature set of our model is fundamentally a combination of the features used in the state-of-the-art joint segmentation and POS tagging model (Zhang and Clark, 2010) and dependency parser (Huang and Sagae, 2010), both of which are used as baseline models in our experiment. However, we must carefully adjust which features are to be activated and when, and how they are combined with which action labels, depending on the type of the features because we intend to perform three tasks in a single incremental framework.

The list of the features used in our joint model is presented in Table 1, where S01–S05, W01–W21, and T01–05 are taken from Zhang and Clark (2010), and P01–P28 are taken from Huang and Sagae (2010). Note that not all features are always considered: each feature is only considered if the action to be performed is included in the list of actions in the “When to apply” column. Because S01–S05 are used to represent the likelihood score of substring sequences, they are only used for A and SH( $t$ ) without being combined with any action label. Because T01–T05 are used to determine the POS tag of the word being shifted, they are only applied for SH( $t$ ). Because W01–W21 are used to determine whether to segment at the current position or not, they are only used for those actions involved in boundary determination decisions (A, SH( $t$ ), RL<sub>0</sub>, and RR<sub>0</sub>). The action labels RL<sub>0</sub>/RR<sub>0</sub> are used to

denote the ‘reduce’ actions that determine the word boundary<sup>2</sup>, whereas RL<sub>1</sub>/RR<sub>1</sub> denote those ‘reduce’ actions that are applied when the word boundary has already been fixed. In addition, to capture the shared nature of boundary determination actions (SH( $t$ ), RL<sub>0</sub>/RR<sub>0</sub>), we use a generalized action label SH’ to represent any of them when combined with W01–W21. We also propose to use the features U01–U03, which we found are effective to adjust the character-level and substring-level scores.

Regarding the parsing features P01–P28, because we found that P01–P17 are also useful for segmentation decisions, these features are applied to all actions including A, with an explicit distinction of action labels RL<sub>0</sub>/RR<sub>0</sub> from RL<sub>1</sub>/RR<sub>1</sub>. On the other hand, P18–P28 are only used when one of the parser actions (SH( $t$ ), RL, or RR) is applied. Note that P07–P09 and P18–P21 (*look-ahead features*) require the look-ahead information of the next word form and POS tags, which cannot be incorporated straightforwardly in an incremental framework. Although we have found that these features can be incorporated using the *delayed features* proposed by Hatori et al. (2011), we did not use them in our current model because it results in the significant increase of computational time.

#### 3.3.1 Dictionary features

Because segmentation using a dictionary alone can serve as a strong baseline in Chinese word segmentation (Sproat et al., 1996), the use of dictionaries is expected to make our joint model more robust and enables us to investigate the contribution of the syntactic dependency in a more realistic setting. Therefore, we optionally use four features D01–D04 associated with external dictionaries. These features distinguish each dictionary source, reflecting the fact that different dictionaries have different characteristics. These features will also be used in our reimplementation of the model by Zhang and Clark (2010).

### 3.4 Adjusting the Learning Rate of Features

In formulating the three tasks in the incremental framework, we found that adjusting the update rate depending on the type of the features (segmentation/tagging vs. parsing) crucially impacts the final performance of the model. To investigate this point, we define the feature vector  $\vec{\phi}$  and score  $\Phi$  of the

<sup>2</sup>A reduce action has an additional effect of fixing the boundary of the top word on the stack if the last action was A or SH( $t$ ).

Id	Feature template	Label	When to apply	
U01	$q_{-1}.e \circ q_{-1}.t$	$\phi$	A, SH( $t$ )	
U02,03	$q_{-1}.e \quad q_{-1}.e \circ q_{-1}.t$	as-is	any	
S01	$q_{-1}.e \circ c_0$	$\phi$	A	
S02	$q_{-1}.t \circ c_0$	$\phi$	A, SH( $t$ )	
S03	$q_{-1}.t \circ q_{-1}.b \circ c_0$	$\phi$	A	
S04	$q_{-1}.t \circ c_0 \circ \mathcal{C}(q_{-1}.b)$	$\phi$	A	
S05	$q_{-1}.t \circ c_0 \circ c_1$	$\phi$	A	
D01	$\text{len}(q_{-1}.w) \circ i$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
D02	$\text{len}(q_{-1}.w) \circ q_{-1}.t \circ i$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
D03	$\text{len}(q_{-1}.w) \circ i$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
D04	$\text{len}(q_{-1}.w) \circ q_{-1}.t \circ i$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
(D01,02: if $q_{-1}.w \in \mathcal{D}_i$ ; D03,04: if $q_{-1}.w \notin \mathcal{D}_i$ )				
W01,02	$q_{-1}.w \quad q_{-2}.w \circ q_{-1}.w$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W03	$q_{-1}.w$ (for single-char word)	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W04	$q_{-1}.b \circ \text{len}(q_{-1}.w)$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W05	$q_{-1}.e \circ \text{len}(q_{-1}.w)$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W06,07	$q_{-1}.e \circ c_0 \quad q_{-1}.b \circ q_{-1}.e$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W08,09	$q_{-1}.w \circ c_0 \quad q_{-2}.e \circ q_{-1}.w$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W10,11	$q_{-1}.b \circ c_0 \quad q_{-2}.e \circ q_{-1}.e$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W12	$q_{-2}.w \circ \text{len}(q_{-1}.w)$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W13	$\text{len}(q_{-2}.w) \circ q_{-1}.w$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W14	$q_{-1}.w \circ q_{-1}.t$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W15	$q_{-2}.t \circ q_{-1}.w$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W16	$q_{-1}.t \circ q_{-1}.w \circ q_{-2}.e$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W17	$q_{-1}.t \circ q_{-1}.w \circ c_0$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W18	$q_{-2}.e \circ q_{-1}.w \circ c_0 \circ q_1.t$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W19	$q_{-1}.t \circ q_{-1}.e$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W20	$q_{-1}.t \circ q_{-1}.e \circ c$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
W21	$q_{-1}.t \circ c \circ \text{cat}(q_{-1}.e)$	A,SH'	A, SH( $t$ ), RR/RL <sub>0</sub>	
(W20, W21: $c \in q_{-1}.w \setminus e$ )				
T01,02	$q_{-1}.t \quad q_{-2}.t \circ q_{-1}.t$	SH( $t$ )	SH( $t$ )	
T03,04	$q_{-1}.w \quad c_0$	SH( $t$ )	SH( $t$ )	
T05	$c_0 \circ q_{-1}.t \circ q_{-1}.e$	SH( $t$ )	SH( $t$ )	
P01,02	$s_0.w \quad s_0.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P03,04	$s_0.w \circ s_0.t \quad s_1.w$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P05,06	$s_1.t \quad s_1.w \circ s_1.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P07,08	$q_0.w \quad q_0.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P09,10	$q_0.w \circ q_0.t \quad s_0.w \circ s_1.w$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P11,12	$s_0.t \circ s_1.t \quad s_0.t \circ q_0.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P13	$s_0.w \circ s_0.t \circ s_1.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P14	$s_0.t \circ s_1.w \circ s_1.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P15	$s_0.w \circ s_1.w \circ s_1.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P16	$s_0.w \circ s_0.t \circ s_1.w$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P17	$s_0.w \circ s_0.t \circ s_1.w \circ s_1.t$	A, SH( $t$ ), RR/RL <sub>0/1</sub>	any	
P18	$s_0.t \circ q_0.t \circ q_1.t$	as-is	SH( $t$ ), RR, RL	
P19	$s_1.t \circ s_0.t \circ q_0.t$	as-is	SH( $t$ ), RR, RL	
P20	$s_0.w \circ q_0.t \circ q_1.t$	as-is	SH( $t$ ), RR, RL	
P21	$s_1.t \circ s_0.w \circ q_0.t$	as-is	SH( $t$ ), RR, RL	
P22	$s_1.t \circ s_1.rc.t \circ s_0.t$	as-is	SH( $t$ ), RR, RL	
P23	$s_1.t \circ s_1.lc.t \circ s_0.t$	as-is	SH( $t$ ), RR, RL	
P24	$s_1.t \circ s_1.rc.t \circ s_0.w$	as-is	SH( $t$ ), RR, RL	
P25	$s_1.t \circ s_1.lc.t \circ s_0.w$	as-is	SH( $t$ ), RR, RL	
P26	$s_1.t \circ s_0.t \circ s_0.rc.t$	as-is	SH( $t$ ), RR, RL	
P27	$s_1.t \circ s_0.w \circ s_0.lc.t$	as-is	SH( $t$ ), RR, RL	
P28	$s_2.t \circ s_1.t \circ s_0.t$	as-is	SH( $t$ ), RR, RL	

\*  $q_{-1}$  and  $q_{-2}$  respectively denote the last-shifted word and the word shifted before  $q_{-1}$ .  $q.w$  and  $q.t$  respectively denote the (root) word form and POS tag of a subtree (word)  $q$ , and  $q.b$  and  $q.e$  the beginning and ending characters of  $q.w$ .  $c_0$  and  $c_1$  are the first and second characters in the queue.  $q.w \setminus e$  denotes the set of characters excluding the ending character of  $q.w$ .  $\text{len}(\cdot)$  denotes the length of the word, capped at 16 if longer.  $\text{cat}(\cdot)$  denotes the category of the character, which is the set of POS tags observed in the training data.  $\mathcal{D}_i$  is a dictionary, a set of words. The action label  $\phi$  means that the feature is not combined with any label; "as-is" denotes the use of the default action set "A, SH( $t$ ), and RR/RL" as is.

Table 1: Feature templates for the full joint model.

	Training		Development			Test		
	#snt	#wrđ	#snt	#wrđ	#oov	#snt	#wrđ	#oov
CTB-5d	16k	438k	804	21k	1.2k	1.9k	50k	3.1k
CTB-5j	18k	494k	352	6.8k	553	348	8.0k	278
CTB-5c	15k	423k	-	-	-	-	-	-
CTB-6	23k	641k	2.1k	60k	3.3k	2.8k	82k	4.6k
CTB-7	31k	718k	10k	237k	13k	10k	245k	13k

Table 2: Statistics of datasets.

action  $a$  being applied to the state  $\psi$  as

$$\Phi(\psi, a) = \vec{\lambda} \cdot \vec{\phi}(\psi, a) = \vec{\lambda} \cdot \left\{ \vec{\phi}_{st}(\psi, a) + \sigma_p \vec{\phi}_p(\psi, a) \right\},$$

where  $\vec{\phi}_{st}$  corresponds to the segmentation and tagging features (those starting with ‘U’, ‘S’, ‘T’, or ‘D’), and  $\vec{\phi}_p$  is the set of the parsing features (starting with ‘P’). Then, if we set  $\sigma_p$  to a number smaller than 1, perceptron updates for the parsing features will be kept small at the early stage of training because the update is proportional to the values of the feature vector. However, even if  $\sigma_p$  is initially small, the global weights for the parsing features will increase as needed and compensate for the small  $\sigma_p$  as the training proceeds. In this way, we can control the contribution of syntactic dependencies at the early stage of training. Section 4.3 shows that the best setting we found is  $\sigma_p = 0.5$ : this result suggests that we probably should resolve remaining errors by preferentially using the local  $n$ -gram based features at the early stage of training. Otherwise, the premature incorporation of the non-local syntactic dependencies might engender overfitting to the training data.

## 4 Experiment

### 4.1 Experimental Settings

We use the Chinese Penn Treebank ver. 5.1, 6.0, and 7.0 (hereinafter CTB-5, CTB-6, and CTB-7) for evaluation. These corpora are split into training, development, and test sets, according to previous works. For CTB-5, we refer to the split by Duan et al. (2007) as CTB-5d, and to the split by Jiang et al. (2008) as CTB-5j. We also prepare a dataset for cross validation: the dataset CTB-5c consists of sentences from CTB-5 excluding the development and test sets of CTB-5d and CTB-5j. We split CTB-5c into five sets (CTB-5c- $n$ ), and alternatively use four of these as the training set and the rest as the test set. CTB-6 is split according to the official split

described in the documentation, and CTB-7 is split according to Wang et al. (2011). The statistics of these splits are shown in Table 2. As external dictionaries, we use the HowNet Word List<sup>3</sup>, consisting of 91,015 words, and page names from the Chinese Wikipedia<sup>4</sup> as of Oct 26, 2011, consisting of 709,352 words. These dictionaries only consist of word forms with no frequency or POS information.

We use standard measures of word-level precision, recall, and F1 score, for evaluating each task. The output of dependencies cannot be correct unless the syntactic head and dependent of the dependency relation are both segmented correctly. Following the standard setting in dependency parsing works, we evaluate the task of dependency parsing with the unlabeled attachment scores excluding punctuations. Statistical significance is tested by McNemar’s test ( $\dagger : p < 0.05$ ,  $\ddagger : p < 0.01$ ).

## 4.2 Baseline and Proposed Models

We use the following baseline and proposed models for evaluation.

- **SegTag**: our reimplementation of the joint segmentation and POS tagging model by Zhang and Clark (2010). Table 5 shows that this reimplementation almost reproduces the accuracy of their implementation. We used the beam of 16, which they reported to achieve the best accuracies.
- **Dep’**: the state-of-the-art dependency parser by Huang and Sagae (2010). We used our reimplementation, which is used in Hatori et al. (2011).
- **Dep**: Dep’ without look-ahead features.
- **TagDep**: the joint POS tagging and dependency parsing model (Hatori et al., 2011), where the look-ahead features are omitted.<sup>5</sup>
- **SegTag+Dep/SegTag+Dep’**: a pipeline combination of SegTag and Dep or Dep’.
- **SegTag+TagDep**: a pipeline combination of SegTag and TagDep, where only the segmentation output of SegTag is used as input to TagDep; the output tags of TagDep are used for evaluation.
- **SegTagDep**: the proposed full joint model.

All of the models described above except Dep’ are based on the same feature sets for segmentation and

<sup>3</sup>[http://www.keenage.com/html/e\\_index.html](http://www.keenage.com/html/e_index.html)

<sup>4</sup><http://zh.wikipedia.org/wiki>

<sup>5</sup>We used the original implementation used in Hatori et al. (2011). In Hatori et al. (2011), we confirmed that omission of the look-ahead features results in a 0.26% decrease in the parsing accuracy on CTB-5d (dev).

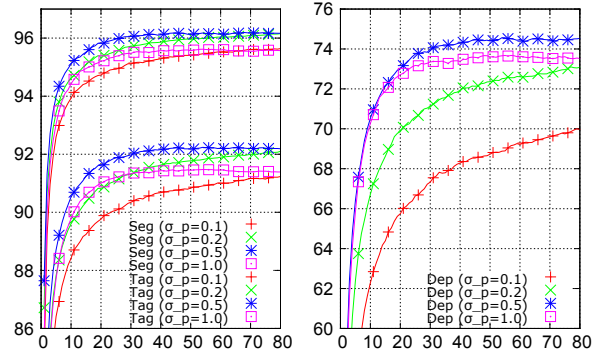


Figure 2: F1 scores (in %) of SegTagDep on CTB-5c-1 w.r.t. the training epoch (x-axis) and parsing feature weights (in legend).

tagging (Zhang and Clark, 2008; Zhang and Clark, 2010) and dependency parsing (Huang and Sagae, 2010). Therefore, we can investigate the contribution of the joint approach through comparison with the pipeline and joint models.

## 4.3 Development Results

We have some parameters to tune: parsing feature weight  $\sigma_p$ , beam size, and training epoch. All these parameters are set based on experiments on CTB-5c. For experiments on CTB-5j, CTB-6, and CTB-7, the training epoch is set using the development set.

Figure 2 shows the F1 scores of the proposed model (SegTagDep) on CTB-5c-1 with respect to the training epoch and different parsing feature weights, where “Seg”, “Tag”, and “Dep” respectively denote the F1 scores of word segmentation, POS tagging, and dependency parsing. In this experiment, the external dictionaries are not used, and the beam size of 32 is used. Interestingly, if we simply set  $\sigma_p$  to 1, the accuracies seem to converge at lower levels. The  $\sigma_p = 0.2$  setting seems to reach almost identical segmentation and tagging accuracies as the best setting  $\sigma_p = 0.5$ , but the convergence occurs more slowly. Based on this experiment, we set  $\sigma_p$  to 0.5 throughout the experiments in this paper.

Table 3 shows the performance and speed of the full joint model (with no dictionaries) on CTB-5c-1 with respect to the beam size. Although even the beam size of 32 results in competitive accuracies for word segmentation and POS tagging, the dependency accuracy is affected most by the increase of the beam size. Based on this experiment, we set the beam size of SegTagDep to 64 throughout the exper-



Beam	Seg	Tag	Dep	Speed
4	94.96	90.19	70.29	5.7
8	95.78	91.53	72.81	3.2
16	96.09	92.09	74.20	1.8
32	96.18	92.24	74.57	0.95
64	96.28	92.37	74.96	0.48

Table 3: F1 scores and speed (in sentences per sec.) of SegTagDep on CTB-5c-1 w.r.t. the beam size.

iments in this paper, unless otherwise noted.

#### 4.4 Main Results

In this section, we present experimentally obtained results using the proposed and baseline models. Table 4 shows the segmentation, POS tagging, and dependency parsing F1 scores of these models on CTB-5c. Irrespective of the existence of the dictionary features, the joint model SegTagDep largely increases the POS tagging and dependency parsing accuracies (by 0.56–0.63% and 2.34–2.44%); the improvements in parsing accuracies are still significant even compared with SegTag+Dep<sup>7</sup> (the pipeline model with the look-ahead features). However, when the external dictionaries are not used (“wo/dict”), no substantial improvements for segmentation accuracies were observed. In contrast, when the dictionaries are used (“w/dict”), the segmentation accuracies are now improved over the baseline model SegTag consistently (on every trial). Although the overall improvement in segmentation is only around 0.1%, more than 1% improvement is observed if we specifically examine OOV<sup>6</sup> words. The difference between “wo/dict” and “w/dict” results suggests that the syntactic dependencies might work as a noise when the segmentation model is insufficiently stable, but the model does improve when it is stable, not receiving negative effects from the syntactic dependencies.

The partially joint model SegTag+TagDep is shown to perform reasonably well in dependency parsing: with dictionaries, it achieved the 2.02% improvement over SegTag+Dep, which is only 0.32% lower than SegTagDep. However, whereas SegTag+TagDep showed no substantial improvement in tagging accuracies over SegTag (when the dictionaries are used), SegTagDep achieved consistent improvements of 0.46% and 0.58% (without/with dic-

<sup>6</sup>We define the OOV words as the words that have not seen in the training data, even when the external dictionaries are used.

System	Seg	Tag
Kruengkrai '09	97.87	93.67
Zhang '10	97.78	93.67
Sun '11	98.17	94.02
Wang '11	98.11	94.18
SegTag	97.66	93.61
SegTagDep	97.73	94.46
SegTag(d)	98.18	94.08
SegTagDep(d)	<b>98.26</b>	<b>94.64</b>

Table 5: Final results on CTB-5j

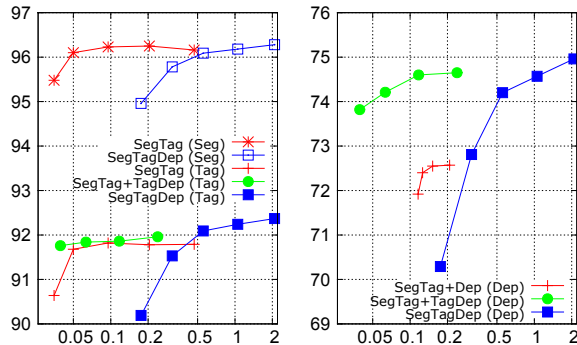


Figure 3: Performance of baseline and joint models w.r.t. the average processing time (in sec.) per sentence. Each point corresponds to the beam size of 4, 8, 16, 32, (64). The beam size of 16 is used for SegTag in SegTag+Dep and SegTag+TagDep.

tionaries); these differences can be attributed to the combination of the relieved error propagation and the incorporation of the syntactic dependencies. In addition, SegTag+TagDep has OOV tagging accuracies consistently lower than SegTag, suggesting that the syntactic dependency has a negative effect on the POS tagging accuracy of OOV words<sup>7</sup>. In contrast, this negative effect is not observed for SegTagDep: both the overall tagging accuracy and the OOV accuracy are improved, demonstrating the effectiveness of the proposed model.

Figure 3 shows the performance and processing time comparison of various models and their combinations. Although SegTagDep takes a few times longer to achieve accuracies comparable to those of SegTag+Dep/TagDep, it seems to present potential

<sup>7</sup>This is consistent with Hatori et al. (2011)’s observation that although the joint POS tagging and dependency parsing improves the accuracy of syntactically influential POS tags, it has a slight side effect of increasing the confusion between general and proper nouns (NN vs. NR).

Model		Segmentation		POS Tagging		Dependency
		ALL	OOV	ALL	OOV	
wo/dict	SegTag+Dep	<b>96.22</b>	72.24	91.74	59.82	72.58
	SegTag+Dep'					72.94 (+0.36 <sup>‡</sup> )
	SegTag+TagDep	96.19 (-0.03)	72.24 (+0.00)	91.86 (+0.12 <sup>‡</sup> )	58.89 (-0.93 <sup>‡</sup> )	74.60 (+2.02 <sup>‡</sup> )
	SegTagDep			<b>92.30</b> (+0.56 <sup>‡</sup> )	<b>61.03</b> (+1.21 <sup>‡</sup> )	<b>74.92</b> (+2.34 <sup>‡</sup> )
w/dict	SegTag+Dep	96.82	78.32	92.34	65.44	73.53
	SegTag+Dep'					73.90 (+0.37 <sup>‡</sup> )
	SegTag+TagDep	<b>96.90</b> (+0.08 <sup>‡</sup> )	<b>79.38</b> (+1.06 <sup>‡</sup> )	92.35 (+0.01)	63.20 (-2.24 <sup>‡</sup> )	75.45 (+1.92 <sup>‡</sup> )
	SegTagDep			<b>92.97</b> (+0.63 <sup>‡</sup> )	<b>67.40</b> (+1.96 <sup>‡</sup> )	<b>75.97</b> (+2.44 <sup>‡</sup> )

Table 4: Segmentation, POS tagging, and (unlabeled attachment) dependency F1 scores averaged over five trials on CTB-5c. Figures in parentheses show the differences over SegTag+Dep (<sup>‡</sup> :  $p < 0.01$ ).

for greater improvement, especially for tagging and parsing accuracies, when a larger beam can be used.

#### 4.5 Comparison with Other Systems

Table 5 and Table 6 show a comparison of the segmentation and POS tagging accuracies with other state-of-the-art models. “Kruengkrai+ ’09” is a lattice-based model by Kruengkrai et al. (2009). “Zhang ’10” is the incremental model by Zhang and Clark (2010). These two systems use no external resources other than the CTB corpora. “Sun+ ’11” is a CRF-based model (Sun, 2011) that uses a combination of several models, with a dictionary of idioms. “Wang+ ’11” is a semi-supervised model by Wang et al. (2011), which additionally uses the Chinese Gigaword Corpus.

Our models with dictionaries (those marked with ‘(d)’) have competitive accuracies to other state-of-the-art systems, and SegTagDep(d) achieved the best reported segmentation and POS tagging accuracies, using no additional corpora other than the dictionaries. Particularly, the POS tagging accuracy is more than 0.4% higher than the previous best system thanks to the contribution of syntactic dependencies. These results also suggest that the use of readily available dictionaries can be more effective than semi-supervised approaches.

## 5 Conclusion

In this paper, we proposed the first joint model for word segmentation, POS tagging, and dependency parsing in Chinese. The model demonstrated substantial improvements on the three tasks over the pipeline combination of the state-of-the-art joint segmentation and POS tagging model, and dependency parser. Particularly, results showed that the

Model	CTB-6 Test			CTB-7 Test		
	Seg	Tag	Dep	Seg	Tag	Dep
Kruengkrai ’09	95.50	90.50	-	95.40	89.86	-
Wang ’11	95.79	91.12	-	95.65	90.46	-
SegTag+Dep	95.46	90.64	72.57	95.49	90.11	71.25
SegTagDep	95.45	91.27	74.88	95.42	90.62	73.58
(diff.)	-0.01	+0.63 <sup>‡</sup>	+2.31 <sup>‡</sup>	-0.07	+0.51 <sup>‡</sup>	+2.33 <sup>‡</sup>
SegTag+Dep(d)	96.13	91.38	73.62	95.98	90.68	72.06
SegTagDep(d)	<b>96.18</b>	<b>91.95</b>	<b>75.76</b>	<b>96.07</b>	<b>91.28</b>	<b>74.58</b>
(diff.)	+0.05	+0.57 <sup>‡</sup>	+2.14 <sup>‡</sup>	+0.09 <sup>‡</sup>	+0.60 <sup>‡</sup>	+2.52 <sup>‡</sup>

Table 6: Final results on CTB-6 and CTB-7

accuracies of POS tagging and dependency parsing were remarkably improved by 0.6% and 2.4%, respectively corresponding to 8.3% and 10.2% error reduction. For word segmentation, although the overall improvement was only around 0.1%, greater than 1% improvements was observed for OOV words. We conducted some comparison experiments of the partially joint and full joint models. Compared to SegTagDep, SegTag+TagDep performs reasonably well in terms of dependency parsing accuracy, whereas the POS tagging accuracies are more than 0.5% lower.

In future work, probabilistic pruning techniques such as the one based on a maximum entropy model are expected to improve the efficiency of the joint model further because the accuracies are apparently still improved if a larger beam can be used. More efficient decoding would also allow the use of the look-ahead features (Hatori et al., 2011) and richer parsing features (Zhang and Nivre, 2011).

**Acknowledgement** We are grateful to the anonymous reviewers for their comments and suggestions, and to Xianchao Wu, Kun Yu, Pontus Stenetorp, and Shin-suke Mori for their helpful feedback.



## References

- Shay B. Cohen and Noah A. Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.
- Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*.
- Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Probabilistic parsing action models for multi-lingual dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.
- Yoav Goldberg and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics (ACL-2008)*.
- Jun Hatori, Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2011. Incremental joint POS tagging and dependency parsing in Chinese. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing (IJCNLP-2011)*.
- Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.
- Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lu. 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Canasai Kruengkrai, Kiyotaka Uchimoto, Jun'ichi Kazama, You Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint Chinese word segmentation and POS tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*.
- Zhenghua Li, Min Zhang, Wanxiang Che, Ting Liu, Wenliang Chen, and Haizhou Haizhou. 2011. Joint models for Chinese POS tagging and dependency parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.
- Xiaoqiang Luo. 2003. A maximum entropy Chinese character-based parser. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*.
- Richard Sproat, Chilin Shih, William Gale, and Nancy Chang. 1996. A stochastic finite-state word-segmentation algorithm for Chinese. *Computational Linguistics*, 22.
- Weiwei Sun. 2011. A stacked sub-word model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- You Wang, Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing (IJCNLP-2011)*.
- Fei Xia. 2000. The part-of-speech tagging guidelines for the Penn Chinese treebank (3.0). Technical Report IRCS-00-07, University of Pennsylvania Institute for Research in Cognitive Science Technical Report, October.
- Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.
- Yue Zhang and Stephen Clark. 2010. A fast decoder for joint word segmentation and POS-tagging using a single discriminative model. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.
- Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (short papers)*.

# Exploring Deterministic Constraints: From a Constrained English POS Tagger to an Efficient ILP Solution to Chinese Word Segmentation

Qiuye Zhao     Mitch Marcus

Dept. of Computer & Information Science  
University of Pennsylvania  
qiuye, mitch@cis.upenn.edu

## Abstract

We show for both English POS tagging and Chinese word segmentation that with proper representation, large number of deterministic constraints can be learned from training examples, and these are useful in constraining probabilistic inference. For tagging, learned constraints are directly used to constrain Viterbi decoding. For segmentation, character-based tagging constraints can be learned with the same templates. However, they are better applied to a word-based model, thus an integer linear programming (ILP) formulation is proposed. For both problems, the corresponding constrained solutions have advantages in both efficiency and accuracy.

## 1 introduction

In recent work, interesting results are reported for applications of integer linear programming (ILP) such as semantic role labeling (SRL) (Roth and Yih, 2005), dependency parsing (Martins et al., 2009) and so on. In an ILP formulation, 'non-local' deterministic constraints on output structures can be naturally incorporated, such as "*a verb cannot take two subject arguments*" for SRL, and the projectivity constraint for dependency parsing. In contrast to probabilistic constraints that are estimated from training examples, this type of constraint is usually hand-written reflecting one's linguistic knowledge.

Dynamic programming techniques based on Markov assumptions, such as Viterbi decoding, cannot handle those 'non-local' constraints as discussed above. However, it is possible to constrain Viterbi

decoding by 'local' constraints, e.g. "*assign label t to word w*" for POS tagging. This type of constraint may come from human input solicited in interactive inference procedure (Kristjansson et al., 2004).

In this work, we explore deterministic constraints for two fundamental NLP problems, English POS tagging and Chinese word segmentation. We show by experiments that, with proper representation, large number of deterministic constraints can be learned automatically from training data, which can then be used to constrain probabilistic inference.

For POS tagging, the learned constraints are directly used to constrain Viterbi decoding. The corresponding constrained tagger is 10 times faster than searching in a raw space pruned with beam-width 5. Tagging accuracy is moderately improved as well.

For Chinese word segmentation (CWS), which can be formulated as character tagging, analogous constraints can be learned with the same templates as English POS tagging. High-quality constraints can be learned with respect to a special tagset, however, with this tagset, the best segmentation accuracy is hard to achieve. Therefore, these character-based constraints are not directly used for determining predictions as in English POS tagging. We propose an ILP formulation of the CWS problem. By adopting this ILP formulation, segmentation F-measure is increased from 0.968 to 0.974, as compared to Viterbi decoding with the same feature set. Moreover, the learned constraints can be applied to reduce the number of possible words over a character sequence, i.e. to reduce the number of variables to set. This reduction of problem size immediately speeds up an ILP solver by more than 100 times.

## 2 English POS tagging

### 2.1 Explore deterministic constraints

Suppose that, following (Chomsky, 1970), we distinguish major lexical categories (Noun, Verb, Adjective and Preposition) by two binary features:  $+|-$  N and  $+|-$  V. Let  $(+N, -V)$ =Noun,  $(-N, +V)$ =Verb,  $(+N, +V)$ =Adjective, and  $(-N, -V)$ =preposition. A word occurring in between a preceding word *the* and a following word *of* always bears the feature  $+N$ . On the other hand, consider the annotation guideline of English Treebank (Marcus et al., 1993) instead. Part-of-speech (POS) tags are used to categorize words, for example, the POS tag VBG tags verbal gerunds, NNS tags nominal plurals, DT tags determiners and so on. Following this POS representation, there are as many as 10 possible POS tags that may occur in between *the-of*, as estimated from the WSJ corpus of Penn Treebank.

#### 2.1.1 Templates of deterministic constraints

To explore determinacy in the distribution of POS tags in Penn Treebank, we need to consider that a POS tag marks the basic syntactic category of a word as well as its morphological inflection. A constraint that may determine the POS category should reflect both the context and the morphological feature of the corresponding word.

The practical difficulty in representing such deterministic constraints is that we do not have a perfect mechanism to analyze morphological features of a word. Endings or prefixes of English words do not deterministically mark their morphological inflections. We propose to compute the morph feature of a word as the set of all of its possible tags, i.e. all tag types that are assigned to the word in training data. Furthermore, we approximate unknown words in testing data by rare words in training data. For a word that occurs less than 5 times in the training corpus, we compute its morph feature as its last two characters, which is also conjoined with binary features indicating whether the rare word contains digits, hyphens or upper-case characters respectively. See examples of morph features in Table 1.

We consider *bigram* and *trigram* templates for generating potentially deterministic constraints. Let  $w_i$  denote the  $i_{th}$  word relative to the current word  $w_0$ ; and  $m_i$  denote the morph feature of  $w_i$ . A

(frequent) $w_0=trades$	(set of possible tags of the word) $m_0=\{NNS, VBZ\}$
(rare) $w_0=time-shares$	(the last two characters...) $m_0=\{-es, HYPHEN\}$

Table 1: Morph features of frequent words and rare words as computed from the WSJ Corpus of Penn Treebank.

bi- -gram	$w_{-1}w_0, w_0w_1, m_{-1}w_0, w_0m_1$ $w_{-1}m_0, m_0w_1, m_{-1}m_0, m_0m_1$
tri- -gram	$w_{-1}w_0w_1, m_{-1}w_0w_1, w_{-1}m_0w_1, m_{-1}m_0w_1$ $w_{-1}w_0m_1, m_{-1}w_0m_1, w_{-1}m_0m_1, m_{-1}m_0m_1$

Table 2: The templates for generating potentially deterministic constraints of English POS tagging.

*bigram* constraint includes one contextual word ( $w_{-1}|w_1$ ) or the corresponding morph feature; and a *trigram* constraint includes both contextual words or their morph features. Each constraint is also conjoined with  $w_0$  or  $m_0$ , as described in Table 2.

#### 2.1.2 Learning of deterministic constraints

In the above section, we explore templates for potentially deterministic constraints that may determine POS category. With respect to a training corpus, if a constraint  $C$  relative to  $w_0$  'always' assigns a certain POS category  $t^*$  to  $w_0$  in its context, i.e.  $\frac{\text{count}(C \wedge t_0=t^*)}{\text{count}(C)} > \text{thr}$ , and this constraint occurs more than a cutoff number, we consider it as a deterministic constraint. The threshold  $\text{thr}$  is a real number just under 1.0 and the cutoff number is empirically set to 5 in our experiments.

#### 2.1.3 Decoding of deterministic constraints

By the above definition, the constraint of  $w_{-1} = \textit{the}$ ,  $m_0 = \{NNS, VBZ\}$  and  $w_1 = \textit{of}$  is deterministic. It determines the POS category of  $w_0$  to be NNS. There are at least two ways of decoding these constraints during POS tagging. Take the word *trades* for example, whose morph feature is  $\{NNS, VBZ\}$ . One alternative is that as long as *trades* occurs between *the-of*, it is tagged with NNS. The second alternative is that the tag decision is made only if all deterministic constraints relative to this occurrence of *trades* agree on the same tag. Both ways of decoding are purely rule-based and involve no probabilistic inference. In favor of a higher precision, we adopt the latter one in our experiments.

raw input	$O(nT^2)$	$n = 23$
The complex financing plan in the S&L bailout law includes...		
constrained input	$O(m_1T + m_2T^2)$	$m_1 = 2, m_2 = 1$
The/DT <b>complex</b> /- <b>financing</b> /- plan/NN in/IN the/DT <b>S&amp;L</b> /- bailout/NN law/NN includes/VBZ ...		

Table 3: Comparison of raw input and constrained input.

## 2.2 Search in a constrained space

Following most previous work, we consider POS tagging as a sequence classification problem and decompose the overall sequence score over the linear structure, i.e.  $\hat{\mathbf{t}} = \arg \max_{\mathbf{t} \in \text{tagGEN}(\mathbf{w})} \sum_{i=1}^n \text{score}(t_i)$  where function **tagGEN** maps input sentence  $\mathbf{w} = w_1 \dots w_n$  to the set of all tag sequences that are of length  $n$ .

If a POS tagger takes raw input only, i.e. for every word, the number of possible tags is a constant  $T$ , the space of **tagGEN** is as large as  $T^n$ . On the other hand, if we decode deterministic constraints first before a probabilistic search, i.e. for some words, the number of possible tags is reduced to 1, the search space is reduced to  $T^m$ , where  $m$  is the number of (unconstrained) words that are not subject to any deterministic constraints.

Viterbi algorithm is widely used for tagging, and runs in  $O(nT^2)$  when searching in an unconstrained space. On the other hand, consider searching in a constrained space. Suppose that among the  $m$  unconstrained words,  $m_1$  of them follow a word that has been tagged by deterministic constraints and  $m_2 (=m-m_1)$  of them follow another unconstrained word. Viterbi decoder runs in  $O(m_1T + m_2T^2)$  while searching in such a constrained space. The example in Table 3 shows raw and constrained input with respect to a typical input sentence.

## Lookahead features

The score of tag predictions are usually computed in a high-dimensional feature space. We adopt the basic feature set used in (Ratnaparkhi, 1996) and (Collins, 2002). Moreover, when deterministic constraints have applied to contextual words of  $w_0$ , it is also possible to include some lookahead feature templates, such as:

$$t_0 \& t_1, t_0 \& t_1 \& t_2, \text{ and } t_{-1} \& t_0 \& t_1$$

where  $t_i$  represents the tag of the  $i_{th}$  word relative

to the current word  $w_0$ . As discussed in (Shen et al., 2007), categorical information of neighbouring words on both sides of  $w_0$  help resolve POS ambiguity of  $w_0$ . In (Shen et al., 2007), lookahead features may be available for use during decoding since searching is bidirectional instead of left-to-right as in Viterbi decoding. In this work, deterministic constraints are decoded before the application of probabilistic models, therefore lookahead features are made available during Viterbi decoding.

## 3 Chinese Word Segmentation (CWS)

### 3.1 Word segmentation as character tagging

Considering the ambiguity problem that a Chinese character may appear in any relative position in a word and the out-of-vocabulary (OOV) problem that it is impossible to observe all words in training data, CWS is widely formulated as a character tagging problem (Xue, 2003). A character-based CWS decoder is to find the highest scoring tag sequence  $\hat{\mathbf{t}}$  over the input character sequence  $\mathbf{c}$ , i.e.

$$\hat{\mathbf{t}} = \arg \max_{\mathbf{t} \in \text{tagGEN}(\mathbf{c})} \sum_{i=1}^n \text{score}(t_i).$$

This is the same formulation as POS tagging. The Viterbi algorithm is also widely used for decoding.

The tag of each character represents its relative position in a word. Two popular tagsets include 1) IB: where B tags the beginning of a word and I all other positions; and 2) BMES: where B, M and E represent the beginning, middle and end of a multi-character word respectively, and S tags a single-character word. For example, after decoding with BMES, 4 consecutive characters associated with the tag sequence BMME compose a word. However, after decoding with IB, characters associated with BIII may compose a word if the following tag is B or only form part of a word if the following tag is I. Even though character tagging accuracy is higher with tagset IB, tagset BMES is more popular in use since better performance of the original problem CWS can be achieved by this tagset.

### Character-based feature templates

We adopt the 'non-lexical-target' feature templates in (Jiang et al., 2008a). Let  $c_i$  denote the  $i_{th}$  character relative to the current character  $c_0$  and  $t_0$

denote the tag assigned to  $c_0$ . The following templates are used:

$c_i \& t_0$  ( $i=-2\dots 2$ ),  $c_i c_{i+1} \& t_0$  ( $i=-2\dots 1$ ) and  $c_{-1} c_1 \& t_0$ .

### Character-based deterministic constraints

We can use the same templates as described in Table 2 to generate potentially deterministic constraints for CWS character tagging, except that there are no morph features computed for Chinese characters. As we will show with experimental results in Section 5.2, useful deterministic constraints for CWS can be learned with tagset IB but not with tagset BMES. It is interesting but not surprising to notice, again, that the determinacy of a problem is sensitive to its representation. Since it is hard to achieve the best segmentations with tagset IB, we propose an indirect way to use these constraints in the following section, instead of applying these constraints as straightforwardly as in English POS tagging.

### 3.2 Word-based word segmentation

A word-based CWS decoder finds the highest scoring segmentation sequence  $\hat{\mathbf{w}}$  that is composed by the input character sequence  $\mathbf{c}$ , i.e.

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w} \in \text{segGEN}(\mathbf{c})} \sum_{i=1}^{|\mathbf{w}|} \text{score}(w_i).$$

where function **segGEN** maps character sequence  $\mathbf{c}$  to the set of all possible segmentations of  $\mathbf{c}$ . For example,  $\mathbf{w} = (c_1 \dots c_{l_1}) \dots (c_{n-l_k+1} \dots c_n)$  represents a segmentation of  $k$  words and the lengths of the first and last word are  $l_1$  and  $l_k$  respectively.

In early work, rule-based models find words one by one based on heuristics such as forward maximum match (Sproat et al., 1996). Exact search is possible with a Viterbi-style algorithm, but beam-search decoding is more popular as used in (Zhang and Clark, 2007) and (Jiang et al., 2008a).

We propose an Integer Linear Programming (ILP) formulation of word segmentation, which is naturally viewed as a word-based model for CWS. Character-based deterministic constraints, as discussed in Section 3.1, can be easily applied.

### 3.3 ILP formulation of CWS

Given a character sequence  $\mathbf{c} = c_1 \dots c_n$ , there are  $s(=n(n+1)/2)$  possible words that are contiguous subsets of  $\mathbf{c}$ , i.e.  $w_1, \dots, w_s \subseteq \mathbf{c}$ . Our goal is to find

n=11	\mathbf{x}	input
raw	55	法正研究从波黑撤军计划
constrained	18	法正研/B究从波/B黑撤/B军计/B划

Table 4: Comparison of raw input and constrained input.

an optimal solution  $\mathbf{x} = x_1 \dots x_s$  that maximizes

$\sum_{i=1}^s \text{score}(w_i) \cdot x_i$ , subject to

$$(1) \sum_{i:c \in w_i} x_i = 1, \forall c \in \mathbf{c};$$

$$(2) x_i \in \{0, 1\}, 1 \leq i \leq s$$

The boolean value of  $x_i$ , as guaranteed by constraint (2), indicates whether  $w_i$  is selected in the segmentation solution or not. Constraint (1) requires every character to be included in exactly one selected word, thus guarantees a proper segmentation of the whole sequence. This resembles the ILP formulation of the set cover problem, though the first constraint is different. Take  $n = 2$  for example, i.e.  $\mathbf{c} = c_1 c_2$ , the set of possible words is  $\{c_1, c_2, c_1 c_2\}$ , i.e.  $s = |\mathbf{x}| = 3$ . There are only two possible solutions subject to constraints (1) and (2),  $\mathbf{x} = 110$  giving an output set  $\{c_1, c_2\}$ , or  $\mathbf{x} = 001$  giving an output set  $\{c_1 c_2\}$ .

The efficiency of solving this problem depends on the number of possible words (contiguous subsets) over a character sequence, i.e. the number of variables in  $\mathbf{x}$ . So as to reduce  $|\mathbf{x}|$ , we apply deterministic constraints predicting IB tags first, which are learned as described in Section 3.1. Possible words are generated with respect to the partially tagged character sequence. A character tagged with  $B$  always occurs at the beginning of a possible word. Table 4 illustrates the constrained and raw input with respect to a typical character sequence.

### 3.4 Character- and word-based features

As studied in previous work, word-based feature templates usually include the word itself, sub-words contained in the word, contextual characters/words and so on. It has been shown that combining the use of character- and word-based features helps improve performance. However, in the character tagging formulation, word-based features are non-local.

To incorporate these non-local features and make the search tractable, various efforts have been made. For example, Jiang et al. (2008a) combine different levels of knowledge in an outside linear model of a two-layer cascaded model; Jiang et al. (2008b) uses the forest re-ranking technique (Huang, 2008); and in (Kruengkrai et al., 2009), only known words in vocabulary are included in the hybrid lattice consisting of both character- and word-level nodes.

We propose to incorporate character-based features in word-based models. Consider a character-based feature function  $\phi(c, t, \mathbf{c})$  that maps a character-tag pair to a high-dimensional feature space, with respect to an input character sequence  $\mathbf{c}$ . For a possible word over  $\mathbf{c}$  of length  $l$ ,  $w_i = c_{i_0} \dots c_{i_0+l-1}$ , tag each character  $c_{i_j}$  in this word with a character-based tag  $t_{i_j}$ . Character-based features of  $w_i$  can be computed as  $\{\phi(c_{i_j}, t_{i_j}, \mathbf{c}) | 0 \leq j < l\}$ . The first row of Table 5 illustrates character-based features of a word of length 3, which is tagged with tagset BMES. From this view, the character-based feature templates defined in Section 3.1 are naturally used in a word-based model.

When character-based features are incorporated into word-based CWS models, some word-based features are no longer of interest, such as the starting character of a word, sub-words contained in the word, contextual characters and so on. We consider word counting features as a complementary to character-based features, following the idea of using web-scale features in previous work, e.g. (Bansal and Klein, 2011). For a possible word  $w$ , let  $count(w)$  return the count of times that  $w$  occurs as a legal word in training data. The word count number is further processed following (Bansal and Klein, 2011),  $wc(w) = \text{floor}(\log(count(w)) * 5) / 5$ . In addition to  $wc(w_i)$ , we also use corresponding word count features of possible words that are composed of the boundary and contextual characters of  $w_i$ . The specific word-based feature templates are illustrated in the second row of Table 5.

## 4 Training

We use the following linear model for scoring predictions:  $score(\mathbf{y}) = \theta^T \phi(\mathbf{x}, \mathbf{y})$ , where  $\phi(\mathbf{y})$  is a high-dimensional binary feature representation of  $\mathbf{y}$  over input  $\mathbf{x}$  and  $\theta$  contains weights of these features. For

character-based	$\phi(c_{i_0}, \mathbf{B}, \mathbf{c}), \phi(c_{i_1}, \mathbf{M}, \mathbf{c}), \phi(c_{i_2}, \mathbf{E}, \mathbf{c})$
word-based	$wc(c_{i_0}c_{i_1}c_{i_2}), wc(c_l c_{i_0}), wc(c_{i_2} c_r)$

Table 5: Character- and word-based features of a possible word  $w_i$  over the input character sequence  $\mathbf{c}$ . Suppose that  $w_i = c_{i_0}c_{i_1}c_{i_2}$ , and its preceding and following characters are  $c_l$  and  $c_r$  respectively.

parameter estimation of  $\theta$ , we use the averaged perceptron as described in (Collins, 2002). This training algorithm relies on the choice of decoding algorithm. When we experiment with different decoders, by default, the parameter weights in use are trained with the corresponding decoding algorithm.

Especially, for experiments with lookahead features of English POS tagging, we prepare training data with the stacked learning technique, in order to alleviate overfitting. More specifically, we divide the training data into  $k$  folds, and tag each fold with the deterministic model learned over the other  $k-1$  folds. The predicted tags of all folds are then merged into the gold training data and used (only) as lookahead features. Sun (2011) uses this technique to merge different levels of predictors for word segmentation.

## 5 Experiments

### 5.1 Data set

We run experiments on English POS tagging on the WSJ corpus in the Penn Treebank. Following most previous work, e.g. (Collins, 2002) and (Shen et al., 2007), we divide this corpus into training set (sections 0-18), development set (sections 19-21) and the final test set (sections 22-24).

We run experiments on Chinese word segmentation on the Penn Chinese Treebank 5.0. Following (Jiang et al., 2008a), we divide this corpus into training set (chapters 1-260), development set (chapters 271-300) and the final test set (chapters 301-325).

### 5.2 Deterministic constraints

Experiments in this section are carried out on the development set. The cutoff number and threshold as defined in 2.1.2, are fixed as 5 and 0.99 respectively.

	precision	recall	$F_1$
<i>bigram</i>	0.993	0.841	0.911
<i>trigram</i>	<b>0.996</b>	0.608	0.755
<i>bi+trigram</i>	0.992	<b>0.857</b>	<b>0.920</b>

Table 6: POS tagging with deterministic constraints. The maximum in each column is bold.

$m_0=\{\text{VBN, VBZ}\}$ & $m_1=\{\text{JJ, VBD, VBN}\}$ $\rightarrow$ VBN
$w_0=\text{also}$ & $m_1=\{\text{VBD, VBN}\}$ $\rightarrow$ RB
$m_0=-\text{es}$ & $m_{-1}=\{\text{IN, RB, RP}\}$ $\rightarrow$ NNS
$w_0=\text{last}$ & $w_{-1}=\text{the}$ $\rightarrow$ JJ

Table 7: Deterministic constraints for POS tagging.

### Deterministic constraints for POS tagging

For English POS tagging, we evaluate the deterministic constraints generated by the templates described in Section 2.1.1. Since these deterministic constraints are only applied to words that occur in a constrained context, we report F-measure as the accuracy measure. Precision  $p$  is defined as the percentage of correct predictions out of all predictions, and recall  $r$  is defined as the percentage of gold predictions that are correctly predicted. F-measure  $F_1$  is computed by  $2pr/(p+r)$ .

As shown in Table 6, deterministic constraints learned with both *bigram* and *trigram* templates are all very accurate in predicting POS tags of words in their context. Constraints generated by *bigram* template alone can already cover 84.1% of the input words with a high precision of 0.993. By adding the constraints generated by *trigram* template, recall is increased to 0.857 with little loss in precision. Since these deterministic constraints are applied before the decoding of probabilistic models, reliably high precision of their predictions is crucial.

There are 114589 *bigram* deterministic constraints and 130647 *trigram* constraints learned from the training data. We show a couple of examples of *bigram* deterministic constraints in Table 7. As defined in Section 2.2, we use the set of all possible POS tags for a word, e.g.  $\{\text{VBN, VBZ}\}$ , as its morph feature if the word is frequent (occurring more than 5 times in training data). For a rare word, the last two characters are used as its morph feature, e.g.  $-\text{es}$ . A constraint is composed of  $w_{-1}$ ,  $w_0$  and  $w_1$ , as well as the morph features  $m_{-1}$ ,  $m_0$  and  $m_1$ . For ex-

tagset	precision	recall	$F_1$
BMES	0.989	0.566	0.720
IB	<b>0.996</b>	<b>0.686</b>	<b>0.812</b>

Table 8: Character tagging with deterministic constraints.

ample, the first constraint in Table 7 determines the tag VBN of  $w_0$ . A deterministic constraint is aware of neither the likelihood of each possible tag or the relative rank of their likelihoods.

### Deterministic constraints for character tagging

For the character tagging formulation of Chinese word segmentation, we discussed two tagsets IB and BMES in Section 3.1. With respect to either tagset, we use both *bigram* and *trigram* templates to generate deterministic constraints for the corresponding tagging problem. These constraints are also evaluated by F-measure as defined above. As shown in Table 8, when tagset IB is used for character tagging, high precision predictions can be made by the deterministic constraints that are learned with respect to this tagset. However, when tagset BMES is used, the learned constraints don't always make reliable predictions, and the overall precision is not high enough to constrain a probabilistic model. Therefore, we will only use the deterministic constraints that predict IB tags in following CWS experiments.

### 5.3 English POS tagging

For English POS tagging, as well as the CWS problem that will be discussed in the next section, we use the development set to choose training iterations (= 5), set beam width etc. The following experiments are done on the final test set.

As introduced in Section 2.2, we adopt a very compact feature set used in (Ratnaparkhi, 1996)<sup>1</sup>. While searching in a constrained space, we can also extend this feature set with some basic lookahead features as defined in Section 2.2. This replicates the feature set B used in (Shen et al., 2007).

In this work, our main interest in the POS tagging problem is on its efficiency. A well-known technique to speed up Viterbi decoding is to conduct beam search. Based on experiments carried out

<sup>1</sup>Our implementation of this feature set is basically the same as the version used in (Collins, 2002).

Ratnaparkhi (1996)’s feature		
	Beam=1	Beam=5
raw	96.46%/3×	97.16/1×
constrained	96.80%/14×	<b>97.20/10×</b>
Feature B in (Shen et al., 2007)		
(Shen et al., 2007)	97.15% (Beam=3)	
constrained	<b>97.03%</b> /11×	<b>97.20/8×</b>

Table 9: POS tagging accuracy and speed. The maximum in each column is bold. The baseline for speed in all cases is the unconstrained tagger using (Ratnaparkhi, 1996)’s feature and conducting a beam (=5) search.

on the development set, we set beam-width of our baseline model as 5. Our baseline model, which uses Ratnaparkhi (1996)’s feature set and conducts a beam (=5) search in the unconstrained space, achieves a tagging accuracy of 97.16%. Tagging accuracy is measured by the percentage of correct predictions out of all gold predictions. We consider the speed of our baseline model as 1×, and compare other taggers with this one. The speed of a POS tagger is measured by the number of input words processed per second.

As shown in Table 9, when the beam-width is reduced from 5 to 1, the tagger (beam=1) is 3 times faster but tagging accuracy is badly hurt. In contrast, when searching in a constrained space rather than the raw space, the constrained tagger (beam=5) is 10 times fast as the baseline and the tagging accuracy is even moderately improved, increasing to 97.20%. When we evaluate the speed of a constrained tagger, the time of decoding deterministic constraints is included. These constraints make more accurate predictions than probabilistic models, thus besides improving the overall tagging speed as we expect, tagging accuracy also improves by a little.

In Viterbi decoding, all possible transitions between two neighbour states are evaluated, so the addition of locally lookahead features may have NO impact on performance. When beam-width is set to 5, tagging accuracy is not improved by the use of Feature B in (Shen et al., 2007); and because the size of the feature model grows, efficiency is hurt. On the other hand, when lookahead features are used, Viterbi-style decoding is less affected by the reduction of beam-width. As compared to the con-

strained greedy tagger using Ratnaparkhi (1996)’s feature set, with the additional use of three locally lookahead feature templates, tagging accuracy is increased from 96.80% to 97.02%.

When no further data is used other than training data, the bidirectional tagger described in (Shen et al., 2007) achieves an accuracy of 97.33%, using a much richer feature set (E) than feature set B, the one we compare with here. As noted above, the addition of three feature templates already has a notable negative impact on efficiency, thus the use of feature set E will hurt tagging efficiency much worse. Rich feature sets are also widely used in other work that pursue state-of-art tagging accuracy, e.g. (Toutanova et al., 2003). In this work, we focus on the most compact feature sets, since tagging efficiency is our main consideration in our work on POS tagging. The proposed constrained taggers as described above can achieve near state-of-art POS tagging accuracy in a much more efficient manner.

#### 5.4 Chinese word segmentation

Like other tagging problems, Viterbi-style decoding is widely used for character tagging for CWS. We transform tagged character sequences to word segmentations first, and then evaluate word segmentations by F-measure, as defined in Section 5.2.

We proposed an ILP formulation of the CWS problem in Section 3.3, where we present a word-based model. In Section 3.4, we describe a way of mapping words to a character-based feature space. From this view, the highest scoring tagging sequence is computed subject to structural constraints, giving us an inference alternative to Viterbi decoding. For example, recall the example of input character sequence  $\mathbf{c} = c_1c_2$  discussed in Section 3.3. The two possible ILP solutions give two possible segmentations  $\{c_1, c_2\}$  and  $\{c_1c_2\}$ , thus there are 2 tag sequences evaluated by ILP, BB and BI. On the other hand, there are 4 tag sequences evaluated by Viterbi decoding: BI, BB, IB and II.

With the same feature templates as described in Section 3.1, we now compare these two decoding methods. Tagset BMES is used for character tagging as well as for mapping words to character-based feature space. We use the same Viterbi decoder as implemented for English POS tagging and use a non-commercial ILP solver included in GNU Linear Pro-



	precision	recall	F-measure
Viterbi	0.971	0.966	0.968
ILP	0.970	0.977	<b>0.974</b>
(Jiang et al., 2008a), POS-			0.971
(Jiang et al., 2008a), POS+			0.973

Table 10: F-measure on Chinese word segmentation. Only character-based features are used. POS-/+ : perceptron trained without/with POS.

gramming Kit (GLPK), version 4.3.<sup>2</sup> As shown in Table 10, optimal solutions returned by an ILP solver are more accurate than optimal solutions returned by a Viterbi decoder. The F-measure is improved by a relative error reduction of 18.8%, from 0.968 to 0.974. These results are compared to the core perceptron trained without POS in (Jiang et al., 2008a). They only report results with ‘lexical-target’ features, a richer feature set than the one we use here. As shown in Table 10, we achieve higher performance even with more compact features.

Joint inference of CWS and Chinese POS tagging is popularly studied in recent work, e.g. (Ng and Low, 2004), (Jiang et al., 2008a), and (Kruengkrai et al., 2009). It has been shown that better performance can be achieved with joint inference, e.g. F-measure 0.978 by the cascaded model in (Jiang et al., 2008a). We focus on the task of word segmentation only in this work and show that a comparable F-measure is achievable in a much more efficient manner. Sun (2011) uses the stacked learning technique to merge different levels of predictors, obtaining a combined system that beats individual ones.

Word-based features can be easily incorporated, since the ILP formulation is more naturally viewed as a word-based model. We extend character-based features with the word count features as described in Section 3.4. Currently, we only use word counts computed from training data, i.e. still a closed test. The addition of these features makes a moderate improvement on the F-measure, from 0.974 to 0.975.

As discussed in Section 3.3, if we are able to determine that some characters always start new words, the number of possible words is reduced, i.e. the number of variables in an ILP solution is reduced. As shown in Table 11, when character se-

<sup>2</sup><http://www.gnu.org/software/glpk>

	F-measure	avg. $ x $	#char per sec.
raw	0.974	1290.4	113 (1 $\times$ )
constrained	0.974	83.75	<b>12190 (107<math>\times</math>)</b>

Table 11: ILP problem size and segmentation speed.

quences are partially tagged by deterministic constraints, the number of possible words per sentence, i.e. avg.  $|x|$ , is reduced from 1290.4 to 83.7. This reduction of ILP problem size has a very important impact on the efficiency. As shown in Table 11, when taking constrained input, the segmentation speed is increased by **107** times over taking raw input, from 113 characters per second to 12,190 characters per second on a dual-core 3.0HZ CPU.

Deterministic constraints predicting IB tags are only used here for constraining possible words. They are very accurate as shown in Section 5.2. Few gold predictions are missed from the constrained set of possible words. As shown in Table 11, F-measure is not affected by applying these constraints, while the efficiency is significantly improved.

## 6 Conclusion and future work

We have shown by experiments that large number of deterministic constraints can be learned from training examples, as long as the proper representation is used. These deterministic constraints are very useful in constraining probabilistic search, for example, they may be directly used for determining predictions as in English POS tagging, or used for reducing the number of variables in an ILP solution as in Chinese word segmentation. The most notable advantage in using these constraints is the increased efficiency. The two applications are both well-studied; there isn’t much space for improving accuracy. Even so, we have shown that as tested with the same feature set for CWS, the proposed ILP formulation significantly improves the F-measure as compared to Viterbi decoding.

These two simple applications suggest that it is of interest to explore data-driven deterministic constraints learnt from training examples. There are more interesting ways in applying these constraints, which we are going to study in future work.

## References

- M. Bansal and D. Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 693–702.
- Noam Chomsky. 1970. Remarks on nominalization. In R Jacobs and P Rosenbaum, editors, *Readings in English Transformational Grammar*, pages 184–221. Ginn.
- Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, EMNLP '02, pages 1–8.
- L. Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*.
- W. Jiang, L. Huang, Q. Liu, and Y. Lü. 2008a. A cascaded linear model for joint chinese word segmentation and part-of-speech tagging. In *In Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*.
- W. Jiang, H. Mi, and Q. Liu. 2008b. Word lattice reranking for chinese word segmentation and part-of-speech tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 385–392.
- T. Kristjansson, A. Culotta, and P. Viola. 2004. Interactive information extraction with constrained conditional random fields. In *In AAAI*, pages 412–418.
- C. Kruengkrai, K. Uchimoto, J. Kazama, Y. Wang, K. Torisawa, and H. Isahara. 2009. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, ACL '09, pages 513–521.
- Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330.
- A. F. T. Martins, N. A. Smith, and E. P. Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 342–350, Singapore.
- H. T. Ng and J. K. Low. 2004. Chinese partof-speech tagging: One-at-a-time or all-at-once? word-based or character-based? In *In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, page 277C284.
- A. Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *In Proceedings of the Empirical Methods in Natural Language Processing Conference (EMNLP)*.
- S. Ravi and K. Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *Proc. ACL*.
- D. Roth and W. Yih. 2005. Integer linear programming inference for conditional random fields. In *In Proceedings of the International Conference on Machine Learning (ICML)*, pages 737–744.
- L. Shen, G. Satta, and A. K. Joshi. 2007. Guided learning for bidirectional sequence classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.
- R. Sproat, W. Gale, C. Shih, and N. Chang. 1996. A stochastic finite-state word-segmentation algorithm for chinese. *Comput. Linguist.*, 22(3):377–404.
- W. Sun. 2011. A stacked sub-word model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of the ACL-HLT 2011*.
- K. Toutanova, D. Klein, C. Manning, and Y. Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL-2003*.
- N. Xue. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, 9(1):29–48.
- Y. Zhang and S. Clark. 2007. Chinese Segmentation with a Word-Based Perceptron Algorithm. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 840–847.

# Author Index

- Abu-Jbara, Amjad, 399  
Adler, Meni, 117  
Asahara, Masayuki, 657
- Bansal, Mohit, 389  
Baroni, Marco, 136  
Barzilay, Regina, 126, 629  
Basili, Roberto, 263  
Berant, Jonathan, 117  
Berg, Alexander, 359  
Berg, Tamara, 359  
Bethard, Steven, 88  
Bisazza, Arianna, 478  
Bittar, André, 730  
Boleda, Gemma, 136  
Boyd-Graber, Jordan, 78  
Branavan, S.R.K., 126  
Bruni, Elia, 136  
Bu, Fan, 449  
Burges, Christopher J.C., 601
- Cabaleiro, Bernardo, 107  
Cancedda, Nicola, 22  
Celikyilmaz, Asli, 330  
Chambers, Nathanael, 98  
Chan, Wen, 582  
Chang, Yi, 611  
Che, Wanxiang, 675  
Chen, Boxing, 930  
Chen, David, 430  
Chen, Wenliang, 213  
Chen, Yidong, 459  
Cheng, Justin, 892  
Chiang, David, 311  
Choi, Yejin, 359  
Chua, Tat-Seng, 582  
Cohen, Shay B., 223  
Collins, Michael, 223  
Constant, Matthieu, 204
- Crespo, Jean-Francois, 611  
Croce, Danilo, 263  
Curran, James R., 497
- Dagan, Ido, 117, 283  
Danescu-Niculescu-Mizil, Cristian, 892  
Dasigi, Pradeep, 399  
Davis, Alexandre, 815  
Demuth, Katherine, 883  
DeNero, John, 146  
Deng, Li, 292  
Diab, Mona, 399, 864  
Diao, Qiming, 536  
Dittmar, Jeremiah, 740  
Dong, Anlei, 611  
Dong, Huailin, 459  
Dragut, Eduard, 997  
Dredze, Mark, 175  
Druck, Gregory, 545  
Duh, Kevin, 1  
Dyer, Chris, 11  
Dymetman, Marc, 22
- Eisenstein, Jacob, 184  
Elsner, Micha, 184  
Engelfriet, Joost, 506
- Falk, Ingrid, 854  
Federico, Marcello, 478  
Felner, Ariel, 283  
Feng, Vanessa Wei, 60  
Feng, Yang, 950  
Foster, Dean P., 223  
Foster, George, 940  
Frank, Michael, 883  
Fraser, Alexander, 469  
Fu, Zhongyang, 526  
Fujino, Akinori, 440

Gamon, Michael, 563  
Gardent, Claire, 592, 854  
Garrido, Guillermo, 107  
Glass, James, 40  
Globerson, Amir, 629  
Goldberger, Jacob, 117  
Goldwater, Sharon, 184, 488  
Green, Spence, 146  
Guerini, Marco, 988  
Guo, Weiwei, 864

Hagège, Caroline, 730  
Hakkani-Tur, Dilek, 330  
Hatori, Jun, 1045  
Hayashi, Katsuhiko, 657  
He, Wei, 979  
He, Xiaodong, 292  
Hirst, Graeme, 60  
Hou, Yufang, 795  
Huang, Eric, 873  
Huang, Liang, 311  
Huang, Zhiheng, 611

Jiang, Jing, 536  
Jiang, Xiao, 1035  
Jin, Ou, 410  
Johansson, Richard, 759  
Johnson, Mark, 488, 883  
Jones, Bevan, 488  
Ju, Qi, 759

Kan, Min-Yen, 1006  
Kanazawa, Makoto, 666  
Kawahara, Tatsuya, 165  
Ke, Dengfeng, 50  
Keerthi, Sathiya, 611  
Keshet, Joseph, 194  
Kessler, Rémy, 730  
Khudanpur, Sanjeev, 175  
Kim, Sungchul, 694  
Klein, Dan, 389, 959  
Kleinberg, Jon, 892  
Klementiev, Alexandre, 647  
Kolachina, Prasanth, 22  
Kolomiyets, Oleksandr, 88  
Konstas, Ioannis, 369  
Korhonen, Anna, 420

Krahmer, Emiel, 1015  
Ku, Hyeonseon, 349  
Kuhn, Roland, 930  
Kushman, Nate, 126  
Kuznetsova, Polina, 359

Laender, Alberto, 815  
Lamirel, Jean-Charles, 854  
Lapata, Mirella, 369, 516  
Larkin, Samuel, 930  
Le Roux, Joseph, 777  
Lee, Chia-ying, 40  
Lee, Lillian, 892  
Lee, Sang-Jo, 1025  
Lei, Tao, 126  
Li, Chi-Ho, 302  
Li, Fangtao, 410  
Li, Haizhou, 213, 902  
Li, Hang, 449  
Li, Mu, 302, 912, 950  
Li, Wenjie, 253  
Li, Xiaoming, 516  
Li, Zhenghua, 675  
Lim, Ee-Peng, 536  
Lin, Shouxun, 750  
Lin, Thomas, 563  
Lin, Ziheng, 1006  
Lippincott, Thomas, 420  
Liu, Bing, 320, 339  
Liu, Chang, 921, 1006  
Liu, Fei, 1035  
Liu, Qiang, 601  
Liu, Qun, 459, 750, 950  
Liu, Shujie, 302  
Liu, Ting, 675, 979  
Liu, Xiaohua, 526, 572  
Liu, Yang, 554  
Livescu, Karen, 194  
Long, Bo, 611  
Lu, Wei, 835

Maletti, Andreas, 506  
Manning, Christopher, 873  
Marcus, Mitch, 1054  
Markert, Katja, 795  
Matsumoto, Yuji, 657

Matsuzaki, Takuya, 1045  
Mauge, Karin, 805  
Mauser, Arne, 156  
Mayfield, Elijah, 740  
McCallum, Andrew, 379, 712  
Meek, Christopher, 601  
Meira Jr., Wagner, 815  
Meng, Weiyi, 997  
Meng, Xinfan, 572  
Minkov, Einat, 845  
Mirroshandel, Seyed Abolghasem, 777  
Miyao, Yusuke, 440, 1045  
Moens, Marie-Francine, 88  
Mooney, Raymond, 349  
Mori, Shinsuke, 165  
Moriceau, Véronique, 730  
Moschitti, Alessandro, 263, 759  
Mukherjee, Arjun, 320, 339

Nagata, Masaaki, 1, 440  
Naidu, Suresh, 740  
Nakagawa, Hiroshi, 721  
Narayan, Shashi, 592  
Naseem, Tahira, 629  
Nasr, Alexis, 777  
Neubig, Graham, 165  
Ney, Hermann, 156  
Ng, Andrew, 873  
Ng, Dominick, 497  
Ng, Hwee Tou, 273, 921, 1006  
Nguyen, Viet-An, 78  
Niu, Feng, 825  
Noh, Tae-Gil, 1025  
Nuhn, Malte, 156

Ó Séaghdha, Diarmuid, 420  
Ordonez, Vicente, 359  
Ozbal, Gozde, 703

Palmer, Martha, 263  
Pan, Sinno Jialin, 410  
Pang, Bo, 545  
Pantel, Patrick, 563  
Park, Seong-Bae, 1025  
Pauls, Adam, 959  
Peñas, Anselmo, 107  
Peng, Xingyuan, 50

Pitler, Emily, 768  
Platt, John C., 601

Qu, Zhonghua, 554

Radev, Dragomir, 399  
Raghavan, Sindhu, 349  
Rastrow, Ariya, 175  
Razmara, Majid, 940  
Ré, Christopher, 825  
Resnik, Philip, 78  
Riedel, Sebastian, 712  
Riezler, Stefan, 11  
Rodrigo, Álvaro, 107  
Rohanimesh, Khash, 805  
Roth, Dan, 835  
Ruvini, Jean-David, 805

Sajjad, Hassan, 469  
Salvati, Sylvain, 666  
Sankaran, Baskaran, 940  
Sarkar, Anoop, 620, 940  
Sato, Issei, 721  
Schmid, Helmut, 469  
Shavlik, Jude, 825  
Shi, Xiaodong, 459  
Shindo, Hiroyuki, 440  
Sigogne, Anthony, 204  
Sim, Khe Chai, 31  
Sim, Yanchuan, 685  
Simianer, Patrick, 11  
Singh, Sameer, 379  
Sistla, Prasad, 997  
Smith, Noah A., 685  
Soares, Altigran, 815  
Socher, Richard, 873  
Son, Jeong-Woo, 1025  
Song, Hyun-Je, 1025  
Stern, Asher, 283  
Stern, Roni, 283  
Stock, Oliviero, 988  
Strapparava, Carlo, 703, 988  
Stratos, Karl, 223  
Strube, Michael, 795  
Su, Jinsong, 459  
Sudoh, Katsuhito, 1  
Sun, Weiwei, 232, 242

Sun, Xu, 253

Takamatsu, Shingo, 721  
Tanaka-Ishii, Kumiko, 969  
Tang, Hao, 194  
Tannier, Xavier, 730  
Titov, Ivan, 647  
Toutanova, Kristina, 694  
Tran, Nam Khanh, 136  
Tsujii, Jun'ichi, 1045  
Tsukada, Hajime, 1

Ungar, Lyle, 223  
Uszkoreit, Hans, 242

van den Bosch, Antal, 1015  
Vaswani, Ashish, 311  
Veloso, Adriano, 815  
Venkatapathy, Sriram, 22

Wan, Xiaojun, 232  
Wang, Haifeng, 459, 979  
Wang, Hong, 997  
Wang, Houfeng, 253, 572  
Wang, Wei, 582  
Wang, William Yang, 740  
Watanabe, Taro, 165, 657  
Watrín, Patrick, 204  
Wei, Furu, 526, 572  
Weng, Fuliang, 1035  
Whitney, Max, 620  
Wick, Michael, 379  
Wilson, Shomir, 638  
Wu, Hua, 459, 979  
Wu, Su-Lin, 611  
Wu, Xianchao, 1  
Wubben, Sander, 1015

Xiao, Xinyan, 750  
Xiong, Deyi, 750, 902  
Xu, Bo, 50  
Xu, Ge, 572  
Xue, Nianwen, 69, 786

Yamaguchi, Hiroshi, 969  
Yan, Rui, 516  
Yang, Nan, 912  
Yang, Qiang, 410

Yang, Yaqin, 786  
Yao, Limin, 712  
Yessenalina, Ainur, 601  
Yogatama, Dani, 685  
Yu, Clement, 997  
Yu, Hwanjo, 694  
Yu, Nenghai, 912

Zettlemoyer, Luke, 845  
Zhang, Ce, 825  
Zhang, Dongdong, 912, 950  
Zhang, Min, 213, 750, 902  
Zhao, Qiuye, 1054  
Zhong, Zhi, 273  
Zhou, Ming, 302, 526, 572  
Zhou, Xiangdong, 582  
Zhou, Xiangyang, 526  
Zhou, Yuping, 69  
Zhu, Feida, 536  
Zhu, Xiaoyan, 410, 449  
Zweig, Geoffrey, 601