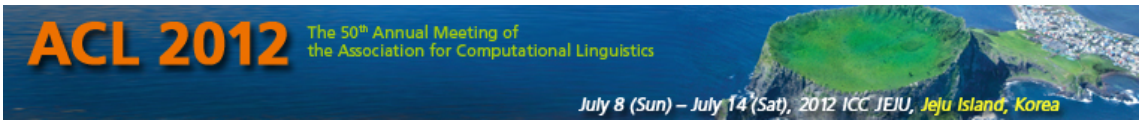
50th Annual Meeting of the Association for Computational Linguistics



Proceedings of the Conference

Volume 2: Short Papers

July 8 - 14, 2012
Jeju Island, Korea

PLATINUM SPONSOR

GOLD SPONSORS

SILVER SPONSORS

BRONZE SPONSORS

SUPPORTERS

SPONSOR FOR BEST PAPER AWARD

IBM Research

Order copies of this and other ACL proceedings from:

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

ISBN 978-1-937284-24-4 (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 Sliver 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 gGmbH

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)
Heuiseok 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 Mellun 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

Jing Bai, Collin Baker, Breck Baldwin, Carmen Banea, Marco Baroni, Regina Barzilay, Roberto Basili, Tilman Becker, Sabine Bergler, Shane Bergsma, Indrajit Bhattacharya, Archana Bhattarai, Jiang Bian, Gann Bierner, Ann Bies, Arianna Bisazza, John Blitzer, Michael Bloodgood, Phil Blunsom, Bernd Bohnet, Ondrej Bojar, Danushka Bollegala, Francis Bond, Kalina Bontcheva, Stefano Borgo, Jordan Boyd-Graber, Kristy Boyer, S.R.K. Branavan, Thorsten Brants, Chris Brew, Ted Briscoe, Samuel Brody, Sabine Buchholz, 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, Sandra Carberry, Claire Cardie, Michael Carl, Xavier Carreras, Francisco Casacuberta, Vittorio Castelli, Asli Celikyilmaz, Mauro Cettolo, Joyce Chai, Nate Chambers, Yee Seng Chan, Ming-Wei Chang, Chia-Ping Chen, John Chen, Keh-Jiann Chen, Xueqi Cheng, Colin Cherry, David Chiang, Hai Leong Chieu, Laura Chiticariu, Yejin Choi, Jennifer Chu-Carroll, Tat-Seng Chua, Ken Church, Alexander Clark, James Clarke, Shay Cohen, Kevyn Collins-Thompson, Marta Ruiz Costa Jussa, Silviu-Petru Cucerzan, Hang Cui, James Cussens

Walter Daelemans, Ido Dagan, R I Damper, Cristian Danescu-Niculescu-Mizil, Van Dang, Laurence Danlos, Dipanjan Das, Hal Daume, Guy De Pauw, Steve DeNeefe, John DeNero, Vera Demberg, Yasuharu Den, Pascal Denis, Markus Dickinson, Mike Dillinger, Xiaowen Ding, Kohji Dohsaka, John Dowding, Doug Downey, Markus Dreyer, Jinhua Du, Kevin Duh, Chris Dyer, Marc Dymetman

Judith Eckle-Kohler, Koji Eguchi, Andreas Eisele, Jacob Eisenstein, Jason Eisner, Michael Elhadad

Benoit Favre, Anna Feldman, Christiane Fellbaum, Raquel Fernandez, Margaret Fleck, Dan Flickinger, Radu Florian, 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, Dmitriy Genzel, Kallirroi Georgila, Sean Gerrish, Daniel Gildea, Jennifer Gillenwater, Dan Gillick, Kevin Gimpel, Roxana Girju, Claudio Giuliano, Amir Globerson, Yoav Goldberg, Sharon Goldwater, Julio Gonzalo, Cyril Goutte, Joao Graca, Agustin Gravano, Spence Green, Charlie Greenbacker, Stephan Greene, Ralph Grishman, Jiafeng Guo, Iryna Gurevych, Adria de Gispert, Josef van Genabith

Barry Haddow, Patrick Haffner, Dilek Hakkani-Tur, David Hall, Keith Hall, Xianpei Han, Jirka Hana, Christian Hardmeier, Kazi Saidul Hasan, Sasa Hasan, Chikara Hashimoto, Ahmed Hassan, Helen Hastie, Katsuhiko Hayashi, Xiaodong He, Zhongjun He, Jeffrey Heinz, John Henderson, Iris Hendrickx, Graeme Hirst, Hieu Hoang, Julia Hockenmaier, 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, Grant Ingersoll, Diana Inkpen, Mitsuru Ishizuka, Tatsuya Izuha, Aranta Diaz de Ilarraza

Jagadeesh Jagarlamudi, Heng Ji, Sittichai Jiampojamarn, 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, Frank Keller, Andre Kempe, 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 Krahmer, Lun-Wei Ku, Sandra Kuebler, Marco Kuhlmann, Roland Kuhn, Seth Kulick, Tom Kwiatkowski, Oi Yee Kwong

Mikel L. Forcada, Wai Lam, Mathias Lambert, Mirella Lapata, Alberto Lavelli, Yoong Keok Lee, Oliver Lemon, Gregor Leusch, Gina-Anne Levow, William Lewis, Fangtao Li, Haizhou Li, Hang Li, Shoushan Li, Wenjie 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, Annie Louis, Xiaofei Lu, Yajuan Lv, Gary Lee

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, Paul McNamee, Arul Menezes, Donald Metzler, Adam Meyers, Jeff Mielke, Rada Mihalcea, Yusuke Miyao, Saif Mohammad, Emad Mohammed, Behrang Mohit, Karo Moilanen, 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 Nassem, Vivi Nastase, Roberto Navigli, 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, Cecilia Ovesdotter Alm

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

Drago Radev, Dragomir Radev, Kira Radinsky, Hema Raghavan, Daniel Ramage, Owen Rambow, Delip Rao, Ari Rappoport, Antoine Raux, Emmanuel Rayner, Roi Reichart, Ehud Reiter, Sebastian Riedel, Jason Riesa, Stefan Riezler, German Rigau, Laura Rimell, Eric Ringger, Alan Ritter, Brian Roark, Horacio Rodríguez, Carolyn Rose, Andrew Rosenberg, Dan Roth, Alexander Rush, Graham Russell

Kenji Sagae, Hasim Sak, Murat Saraclar, Anoop Sarkar, Sudeshna Sarkar, David Schlangen,

Helmut Schmid, Nathan Schneider, William Schuler, Sabine Schulte im Walde, Hinrich Schütze, Satoshi Sekine, Violeta Seretan, Hendra Setiawan, Dipti Sharma, Libin Shen, Wade Shen, Shuming Shi, Eyal Shnarch, Candy Sidner, Michel Simard, Gabriel Skantze, Rion Snow, Stephen Soderland, Yang Song, Youngin Song, Lucia Specia, Rohini Srihari, Manfred Stede, 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, Diarmuid ó Séaghdha

Hiroya Takamura, Koichi Takeda, Partha Talukdar, Kumiko Tanaka-Ishii, Stefan Thater, Mariet Theune, Joerg Tiedemann, Christoph Tillmann, Ivan Titov, Takenobu Tokunaga, 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, Marc Vilain, David Vilar, Aline Villavicencio, Sami Virpioja, Andreas Vlachos, Piek Vossen

Marilyn Walker, 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, Tak-Lam Wong, Kristian Woodsend, Chung-Hsien Wu, Xianchao Wu

Fei Xia, Yunqing Xia, Lei Xie, Shasha Xie, Deyi Xiong, Gu Xu, Peng Xu, Nianwen 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, Liang-Chih Yu, Yisong Yue, Deniz Yuret

Fabio Zanzotto, Jakub Zavrel, Klaus Zechner, Dmitry Zelenko, 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, Xiaodan Zhu, Michael Zock, Ingrid Zukerman

Table of Contents

Conference Program

Monday July 9, 2012

(18:00 – 20:30) Poster Session

Wednesday July 11, 2012

Session 7a: (11:00 – 12:30) Parsing

Higher-order Constituent Parsing and Parser Combination
Xiao Chen and Chunyu Kit

Joint Evaluation of Morphological Segmentation and Syntactic Parsing
Reut Tsarfaty, Joakim Nivre and Evelina Andersson

A Comparison of Chinese Parsers for Stanford Dependencies
Wanxiang Che, Valentin Spitkovsky and Ting Liu

A Feature-Rich Constituent Context Model for Grammar Induction
Dave Golland, John DeNero and Jakob Uszkoreit

Session 7b: (11:00 – 12:30) Machine Translation

Private Access to Phrase Tables for Statistical Machine Translation
Nicola Cancedda

Fast and Scalable Decoding with Language Model Look-Ahead for Phrase-based Statistical Machine Translation
Joern Wuebker, Hermann Ney and Richard Zens

Head-Driven Hierarchical Phrase-based Translation
Junhui Li, Zhaopeng Tu, Guodong Zhou and Josef van Genabith

Joint Learning of a Dual SMT System for Paraphrase Generation
Hong Sun and Ming Zhou

Wednesday July 11, 2012 (continued)

Session 7c: (11:00 – 12:30) Relations and events

A Novel Burst-based Text Representation Model for Scalable Event Detection
Xin Zhao, Rishan Chen, Kai Fan, Hongfei Yan and Xiaoming Li

A Graph-based Cross-lingual Projection Approach for Weakly Supervised Relation Extraction
Seokhwan Kim and Gary Geunbae Lee

Pattern Learning for Relation Extraction with a Hierarchical Topic Model
Enrique Alfonseca, Katja Filippova, Jean-Yves Delort and Guillermo Garrido

Session 7d: (11:00 – 12:30) Discourse

Self-Disclosure and Relationship Strength in Twitter Conversations
JinYeong Bak, Suin Kim and Alice Oh

Genre Independent Subgroup Detection in Online Discussion Threads: A Study of Implicit Attitude using Textual Latent Semantics
Pradeep Dasigi, Weiwei Guo and Mona Diab

Learning to Temporally Order Medical Events in Clinical Text
Preethi Raghavan, Albert Lai and Eric Fosler-Lussier

A Context-sensitive, Multi-faceted Model of Lexico-Conceptual Affect
Tony Veale

Session 7e: (11:00 – 12:30) Machine Learning

Decoding Running Key Ciphers
Sravana Reddy and Kevin Knight

Using Rejuvenation to Improve Particle Filtering for Bayesian Word Segmentation
Benjamin Börschinger and Mark Johnson

Baselines and Bigrams: Simple, Good Sentiment and Topic Classification
Sida Wang and Christopher Manning

Wednesday July 11, 2012 (continued)

Automatically Learning Measures of Child Language Development
Sam Sahakian and Benjamin Snyder

Session 8a: (16:00 – 17:30) Parsing and Machine Translation

A Comparative Study of Target Dependency Structures for Statistical Machine Translation
Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada and Masaaki Nagata

Robust Conversion of CCG Derivations to Phrase Structure Trees
Jonathan K. Kummerfeld, Dan Klein and James R. Curran

Estimating Compact Yet Rich Tree Insertion Grammars
Elif Yamangil and Stuart Shieber

Topic Models for Dynamic Translation Model Adaptation
Vladimir Eidelman, Jordan Boyd-Graber and Philip Resnik

Session 8b: (16:00 – 17:30) Multilinguality

Detecting Semantic Equivalence and Information Disparity in Cross-lingual Documents
Yashar Mehdad, Matteo Negri and Marcello Federico

Cross-lingual Parse Disambiguation based on Semantic Correspondence
Lea Frermann and Francis Bond

Learning to Find Translations and Transliterations on the Web
Joseph Z. Chang, Jason S. Chang and Roger Jyh-Shing Jang

Beefmoves: Dissemination, Diversity, and Dynamics of English Borrowings in a German Hip Hop Forum
Matt Garley and Julia Hockenmaier

Wednesday July 11, 2012 (continued)

Session 8c: (16:00 – 17:30) Meaning

Learning the Latent Semantics of a Concept from its Definition
Weiwei Guo and Mona Diab

Unsupervised Semantic Role Induction with Global Role Ordering
Nikhil Garg and James Henserdon

Humor as Circuits in Semantic Networks
Igor Labutov and Hod Lipson

Crowdsourcing Inference-Rule Evaluation
Naomi Zeichner, Jonathan Berant and Ido Dagan

Session 8d: (16:00 – 17:30) NLP Apps and data

A Comprehensive Gold Standard for the Enron Organizational Hierarchy
Apoorv Agarwal, Adinoyi Omuya, Aaron Harnly and Owen Rambow

A Two-step Approach to Sentence Compression of Spoken Utterances
Dong Wang, Xian Qian and Yang Liu

Syntactic Stylometry for Deception Detection
Song Feng, Ritwik Banerjee and Yejin Choi

Transforming Standard Arabic to Colloquial Arabic
Emad Mohamed, Behrang Mohit and Kemal Oflazer

Wednesday July 11, 2012 (continued)

Session 8e: (16:00 – 17:30) NLP Apps

Corpus-based Interpretation of Instructions in Virtual Environments
Luciana Benotti, Martin Villalba, Tessa Lau and Julian Cerruti

Automatically Mining Question Reformulation Patterns from Search Log Data
Xiaobing Xue, Yu Tao, Daxin Jiang and Hang Li

Native Language Detection with Tree Substitution Grammars
Benjamin Swanson and Eugene Charniak

Tense and Aspect Error Correction for ESL Learners Using Global Context
Toshikazu Tajiri, Mamoru Komachi and Yuji Matsumoto

Monday July 9, 2012

(18:00 – 20:30) Poster Session

Movie-DiC: a Movie Dialogue Corpus for Research and Development
Rafael E. Banchs

Combining Textual Entailment and Argumentation Theory for Supporting Online Debates Interactions
Elena Cabrio and Serena Villata

Towards the Unsupervised Acquisition of Discourse Relations
Christian Chiarcos

Arabic Retrieval Revisited: Morphological Hole Filling
Kareem Darwish and Ahmed Ali

Extracting and modeling durations for habits and events from Twitter
Jennifer Williams and Graham Katz

Event Linking: Grounding Event Reference in a News Archive
Joel Nothman, Matthew Honnibal, Ben Hachey and James R. Curran

Monday July 9, 2012 (continued)

Coupling Label Propagation and Constraints for Temporal Fact Extraction
Yafang Wang, Maximilian Dylla, Marc Spaniol and Gerhard Weikum

Using Search-Logs to Improve Query Tagging
Kuzman Ganchev, Keith Hall, Ryan McDonald and Slav Petrov

Toward Automatically Assembling Hittite-Language Cuneiform Tablet Fragments into Larger Texts
Stephen Tyndall

A Corpus of Textual Revisions in Second Language Writing
John Lee and Jonathan Webster

Coarse Lexical Semantic Annotation with Supersenses: An Arabic Case Study
Nathan Schneider, Behrang Mohit, Kemal Oflazer and Noah A. Smith

Word Epoch Disambiguation: Finding How Words Change Over Time
Rada Mihalcea and Vivi Nastase

Authorship Attribution with Author-aware Topic Models
Yanir Seroussi, Fabian Bohnert and Ingrid Zukerman

Information-theoretic Multi-view Domain Adaptation
Pei Yang, Wei Gao, Qi Tan and Kam-Fai Wong

Efficient Tree-Based Topic Modeling
Yuening Hu and Jordan Boyd-Graber

Learning Better Rule Extraction with Translation Span Alignment
Jingbo Zhu, Tong Xiao and Chunliang Zhang

Enhancing Statistical Machine Translation with Character Alignment
Ning Xi, Guangchao Tang, Xinyu Dai, Shujian Huang and Jiajun Chen

Translation Model Size Reduction for Hierarchical Phrase-based Statistical Machine Translation
Seung-Wook Lee, Dongdong Zhang, Mu Li, Ming Zhou and Hae-Chang Rim

Monday July 9, 2012 (continued)

Heuristic Cube Pruning in Linear Time
Andrea Gesmundo, Giorgio Satta and James Henderson

Combining Word-Level and Character-Level Models for Machine Translation Between Closely-Related Languages
Preslav Nakov and Jörg Tiedemann

Improving the IBM Alignment Models Using Variational Bayes
Darcey Riley and Daniel Gildea

Post-ordering by Parsing for Japanese-English Statistical Machine Translation
Isao Goto, Masao Utiyama and Eiichiro Sumita

An Exploration of Forest-to-String Translation: Does Translation Help or Hurt Parsing?
Hui Zhang and David Chiang

Unsupervised Morphology Rivals Supervised Morphology for Arabic MT
David Stallard, Jacob Devlin, Michael Kayser, Yoong Keok Lee and Regina Barzilay

A Meta Learning Approach to Grammatical Error Correction
Hongsuck Seo, Jonghoon Lee, Seokhwan Kim, Kyusong Lee, Sechun Kang and Gary Geunbae Lee

Fine Granular Aspect Analysis using Latent Structural Models
Lei Fang and Minlie Huang

Identifying High-Impact Sub-Structures for Convolution Kernels in Document-level Sentiment Classification
Zhaopeng Tu, Yifan He, Jennifer Foster, Josef van Genabith, Qun Liu and Shouxun Lin

Exploiting Latent Information to Predict Diffusions of Novel Topics on Social Networks
Tsung-Ting Kuo, San-Chuan Hung, Wei-Shih Lin, Nanyun Peng, Shou-De Lin and Wei-Fen Lin

Sentence Compression with Semantic Role Constraints
Katsumasa Yoshikawa, Ryu Iida, Tsutomu Hirao and Manabu Okumura

Fully Abstractive Approach to Guided Summarization
Pierre-Etienne Genest and Guy Lapalme

Monday July 9, 2012 (continued)

Assessing the Effect of Inconsistent Assessors on Summarization Evaluation
Karolina Owczarzak, Peter A. Rankel, Hoa Trang Dang and John M. Conroy

Fast and Robust Part-of-Speech Tagging Using Dynamic Model Selection
Jinho D. Choi and Martha Palmer

Lemmatisation as a Tagging Task
Andrea Gesmundo and Tanja Samardzic

How Are Spelling Errors Generated and Corrected? A Study of Corrected and Uncorrected Spelling Errors Using Keystroke Logs
Yukino Baba and Hisami Suzuki

Tokenization: Returning to a Long Solved Problem — A Survey, Contrastive Experiment, Recommendations, and Toolkit —
Rebecca Dridan and Stephan Oepen

Unsupervized Word Segmentation: the Case for Mandarin Chinese
Pierre Magistry and Benoît Sagot

Grammar Error Correction Using Pseudo-Error Sentences and Domain Adaptation
Kenji Imamura, Kuniko Saito, Kugatsu Sadamitsu and Hitoshi Nishikawa

Higher-Order Constituent Parsing and Parser Combination*

Xiao Chen and Chunyu Kit
Department of Chinese, Translation and Linguistics
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong SAR, China
{cxiao2,ctckit}@cityu.edu.hk

Abstract

This paper presents a higher-order model for constituent parsing aimed at utilizing more local structural context to decide the score of a grammar rule instance in a parse tree. Experiments on English and Chinese treebanks confirm its advantage over its first-order version. It achieves its best F1 scores of 91.86% and 85.58% on the two languages, respectively, and further pushes them to 92.80% and 85.60% via combination with other high-performance parsers.

1 Introduction

Factorization is crucial to discriminative parsing. Previous discriminative parsing models usually factor a parse tree into a set of parts. Each part is scored separately to ensure tractability. In dependency parsing (DP), the number of dependencies in a part is called the *order* of a DP model (Koo and Collins, 2010). Accordingly, existing graph-based DP models can be categorized into tree groups, namely, the first-order (Eisner, 1996; McDonald et al., 2005a; McDonald et al., 2005b), second-order (McDonald and Pereira, 2006; Carreras, 2007) and third-order (Koo and Collins, 2010) models.

Similarly, we can define the *order* of constituent parsing in terms of the number of grammar rules in a part. Then, the previous discriminative constituent parsing models (Johnson, 2001; Henderson, 2004; Taskar et al., 2004; Petrov and Klein, 2008a; Petrov and Klein, 2008b; Finkel et al., 2008) are the first-order ones, because there is only one grammar rule in a part. The discriminative re-scoring models (Collins, 2000; Collins and Duffy, 2002; Charniak and Johnson, 2005; Huang, 2008) can be viewed as previous attempts to higher-order constituent parsing, using some parts containing more than one grammar rule as non-local features.

In this paper, we present a higher-order constituent parsing model[1] based on these previous works. It allows multiple adjacent grammar rules in each part of a parse tree, so as to utilize more local structural context to decide the plausibility of a grammar rule instance. Evaluated on the PTB WSJ and Chinese Treebank, it achieves its best F1 scores of 91.86% and 85.58%, respectively. Combined with other high-performance parsers under the framework of constituent recombination (Sagae and Lavie, 2006; Fossum and Knight, 2009), this model further enhances the F1 scores to 92.80% and 85.60%, the highest ones achieved so far on these two data sets.

2 Higher-order Constituent Parsing

Discriminative parsing is aimed to learn a function $f : \mathcal{S} \rightarrow \mathcal{T}$ from a set of sentences $\mathcal{S}$ to a set of valid parses $\mathcal{T}$ according to a given CFG, which maps an input sentence $s \in \mathcal{S}$ to a set of candidate parses $\mathcal{T}(s)$. The function takes the following discriminative form:

$$f(s) = \arg\max_{t \in \mathcal{T}(s)} g(t, s) \quad (1)$$

*The research reported in this paper was partially supported by the Research Grants Council of HKSAR, China, through the GRF Grant 9041597 (CityU 144410).

[1] http://code.google.com/p/gazaparser/

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 1–5,
Jeju, Republic of Korea, 8-14 July 2012.

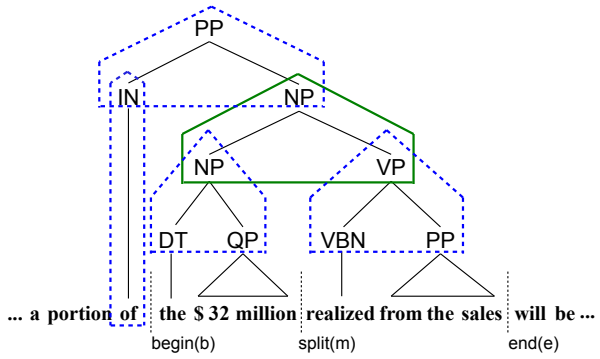


Figure 1: A part of a parse tree centered at NP → NP VP

where $g(t, s)$ is a scoring function to evaluate the event that t is the parse of s. Following Collins (2002), this scoring function is formulated in the linear form

$$g(t, s) = \theta \cdot \Psi(t, s), \tag{2}$$

where $\Psi(t, s)$ is a vector of features and θ the vector of their associated weights. To ensure tractability, this model is factorized as

$$g(t, s) = \sum_{r \in t} g(\mathcal{Q}(r), s) = \sum_{r \in t} \theta \cdot \Phi(\mathcal{Q}(r), s), \tag{3}$$

where $g(\mathcal{Q}(r), s)$ scores $\mathcal{Q}(r)$, a part centered at grammar rule instance r in t, and $\Phi(\mathcal{Q}(r), s)$ is the vector of features for $\mathcal{Q}(r)$. Each $\mathcal{Q}(r)$ makes its own contribution to $g(t, s)$. A part in a parse tree is illustrated in Figure 1. It consists of the center grammar rule instance NP → NP VP and a set of immediate neighbors, i.e., its parent PP → IN NP, its children NP → DT QP and VP → VBN PP, and its sibling IN → of. This set of neighboring rule instances forms a local structural context to provide useful information to determine the plausibility of the center rule instance.

2.1 Feature

The feature vector $\Phi(\mathcal{Q}(r), s)$ consists of a series of features $\{\phi_i(\mathcal{Q}(r), s))|i \geq 0\}$. The first feature $\phi_0(\mathcal{Q}(r), s)$ is calculated with a PCFG-based generative parsing model (Petrov and Klein, 2007), as defined in (4) below, where r is the grammar rule instance $\mathcal{A} \rightarrow \mathcal{B}\ \mathcal{C}$ that covers the span from the b-th to the e-th word, splitting at the m-th word, x, y and z are latent variables in the PCFG-based model, and $I(\cdot)$ and $O(\cdot)$ are the inside and outside probabilities, respectively.

All other features $\phi_i(\mathcal{Q}(r), s)$ are binary functions that indicate whether a configuration exists in $\mathcal{Q}(r)$ and s. These features are by their own nature in two categories, namely, lexical and structural. All features extracted from the part in Figure 1 are demonstrated in Table 1. Some back-off structural features are used for smoothing, which cannot be presented due to limited space. With only lexical features in a part, this parsing model backs off to a first-order one similar to those in the previous works. Adding structural features, each involving a least a neighboring rule instance, makes it a higher-order parsing model.

2.2 Decoding

The factorization of the parsing model allows us to develop an exact decoding algorithm for it. Following Huang (2008), this algorithm traverses a parse forest in a bottom-up manner. However, it determines and keeps the best derivation for every grammar rule instance instead of for each node. Because all structures above the current rule instance is not determined yet, the computation of its nonlocal structural features, e.g., parent and sibling features, has to be delayed until it joins an upper level structure. For example, when computing the score of a derivation under the center rule NP → NP VP in Figure 1, the algorithm will extract child features from its children NP → DT QP and VP → VBN PP. The parent and sibling features of the two child rules can also be extracted from the current derivation and used to calculate the score of this derivation. But parent and sibling features for the center rule will not be computed until the decoding process reaches the rule above, i.e., PP → IN NP.

This algorithm is more complex than the approximate decoding algorithm of Huang (2008). However, its efficiency heavily depends on the size of the parse forest it has to handle. Forest pruning (Char-

$$\phi_0(\mathcal{Q}(r), s) = \frac{\sum_x \sum_y \sum_z O(\mathcal{A}_x, b, e)\mathcal{P}(\mathcal{A}_x \rightarrow \mathcal{B}_y\ \mathcal{C}_z)I(\mathcal{B}_y, b, m)I(\mathcal{C}_z, m, e)}{I(\mathcal{S}, 0, n)} \tag{4}$$

Template		Description		Comments
Lexical feature	N-gram on inner /outer edge	$w_{b/e+l}$(l=0,1,2,3,4)	& b/e & l & NP	Similar to the *distributional similarity cluster bigrams* features in Finkel et al. (2008)
		$w_{b/e-l}$(l=1,2,3,4,5)	& b/e & l & NP	
		$w_{b/e+l}w_{b/e+l+1}$(l=0,1,2,3)	& b/e & l & NP	
		$w_{b/e-l-1}w_{b/e-l}$(l=1,2,3,4)	& b/e & l & NP	
		$w_{b/e+l}w_{b/e+l+1}w_{b/e+l+2}$($l$=0,1,2)	& b/e & l & NP	
		$w_{b/e-l-2}w_{b/e-l-1}w_{b/e-l}$($l$=1,2,3)	& b/e & l & NP	
	Bigram on edges	$w_{b/e-1}w_{b/e}$	& NP	Similar to the lexical span features in Taskar et al. (2004) and Petrov and Klein (2008b)
	Split pair	$w_{m-1}w_m$	& NP → NP VP	
	Inner/Outer pair	$w_b w_{e-1}$	& NP → NP VP	
		$w_{b-1}w_e$	& NP → NP VP	
	Rule bigram	Left & NP	& NP	Similar to the *bigrams* features in Collins (2000)
		Right & NP	& NP	
Structural feature	Parent	PP → IN NP	& NP → NP VP	Similar to the *grandparent rules* features in Collins (2000)
	Child	NP → DT QP & VP → VBN PP	& NP → NP VP	
		NP → DT QP	& NP → NP VP	
		VP → VBN PP	& NP → NP VP	
	Sibling	Left & IN → of	& NP → NP VP	

Table 1: Examples of lexical and structural feature

niak and Johnson, 2005; Petrov and Klein, 2007) is therefore adopted in our implementation for efficiency enhancement. A parallel decoding strategy is also developed to further improve the efficiency without loss of optimality. Interested readers can refer to Chen (2012) for more technical details of this algorithm.

3 Constituent Recombination

Following Fossum and Knight (2009), our constituent weighting scheme for parser combination uses multiple outputs of independent parsers. Suppose each parser generates a k-best parse list for an input sentence, the weight of a candidate constituent c is defined as

$$\omega(c) = \sum_i \sum_k \lambda_i \delta(c, t_{i,k}) f(t_{i,k}), \qquad (5)$$

where i is the index of an individual parser, λ_i the weight indicating the confidence of a parser, $\delta(c, t_{i,k})$ a binary function indicating whether c is contained in $t_{i,k}$, the k-th parse output from the i-th parser, and $f(t_{i,k})$ the score of the k-th parse assigned by the i-th parser, as defined in Fossum and Knight (2009).

The weight of a recombined parse is defined as the sum of weights of all constituents in the parse. However, this definition has a systematic bias towards selecting a parse with as many constituents as possible for the highest weight. A pruning threshold ρ, similar to the one in Sagae and Lavie (2006), is therefore needed to restrain the number of constituents in a recombined parse. The parameters λ_i and ρ are tuned by the Powell's method (Powell, 1964) on a development set, using the F1 score of PARSEVAL (Black et al., 1991) as objective.

	English	Chinese
Train.	Section 2-21	Art. 1-270,400-1151
Dev.	Section 22/24	Art. 301-325
Test.	Section 23	Art. 271-300

Table 2: Experiment Setup

4 Experiment

Our parsing models are evaluated on both English and Chinese treebanks, i.e., the WSJ section of Penn Treebank 3.0 (LDC99T42) and the Chinese Treebank 5.1 (LDC2005T01U01). In order to compare with previous works, we opt for the same split as in Petrov and Klein (2007), as listed in Table 2. For parser combination, we follow the setting of Fossum and Knight (2009), using Section 24 instead of Section 22 of WSJ treebank as development set.

In this work, the lexical model of Chen and Kit (2011) is combined with our syntactic model under the framework of product-of-experts (Hinton, 2002). A factor λ is introduced to balance the two models. It is tuned on a development set using the gold sec-

	English			Chinese		
	R(%)	P(%)	F1(%)	R(%)	P(%)	F1(%)
Berkeley parser	89.71	90.03	*89.87*	82.00	84.48	*83.22*
First-order	91.33	91.79	91.56	84.14	86.23	85.17
Higher-order	91.62	92.11	91.86	84.24	86.54	85.37
Higher-order+λ	91.60	92.13	**91.86**	84.45	86.74	**85.58**
Stanford parser	-	-	-	77.40	79.57	78.47
C&J parser	91.04	91.76	91.40	-	-	-
Conbination	92.02	93.60	**92.80**	82.44	89.01	**85.60**

Table 3: The performance of our parsing models on the English and Chinese test sets.

System	F1(%)	EX(%)
Single		
Charniak (2000)	89.70	
Berkeley parser	89.87	36.7
Bod (2003)	90.70	
Carreras et al. (2008)	91.1	
Re-scoring		
Collins (2000)	89.70	
Charniak and Johnson (2005)	91.02	
The parser of Charniak and Johnson	91.40	43.54
Huang (2008)	91.69	43.5
Combination		
Fossum and Knight (2009)	92.4	
Zhang et al. (2009)	92.3	
Petrov (2010)	91.85	41.9
Self-training		
Zhang et al. (2009) (s.t.+combo)	92.62	
Huang et al. (2010) (single)	91.59	40.3
Huang et al. (2010) (combo)	92.39	43.1
Our single	91.86	40.89
Our combo	**92.80**	41.60

Table 4: Performance comparison on the English test set

System	F1(%)	EX(%)
Single		
Charniak (2000)	80.85	
Stanford parser	78.47	26.44
Berkeley parser	83.22	31.32
Burkett and Klein (2008)	84.24	
Combination		
Zhang et al. (2009) (combo)	85.45	
Our single	85.56	31.61
Our combo	**85.60**	29.02

Table 5: Performance comparison on the Chinese test set

tion search algorithm (Kiefer, 1953). The parameters θ of each parsing model are estimated from a training set using an averaged perceptron algorithm, following Collins (2002) and Huang (2008).

The performance of our *first-* and *higher-order* parsing models on all sentences of the two test sets is presented in Table 3, where λ indicates a tuned balance factor. This parser is also combined with the parser of Charniak and Johnson (2005)[2] and the Stanford. parser[3] The best combination results in Table 3 are achieved with k=70 for English and k=100 for Chinese for selecting the k-best parses. Our results are compared with the best previous ones on the same test sets in Tables 4 and 5. All scores listed in these tables are calculated with `evalb`,[4] and EX is the *complete match rate*.

5 Conclusion

This paper has presented a higher-order model for constituent parsing that factorizes a parse tree into larger parts than before, in hopes of increasing its power of discriminating the true parse from the others without losing tractability. A performance gain of 0.3%-0.4% demonstrates its advantage over its first-order version. Including a PCFG-based model as its basic feature, this model achieves a better performance than previous single and re-scoring parsers, and its combination with other parsers performs even better (by about 1%). More importantly, it extends the existing works into a more general framework of constituent parsing to utilize more lexical and structural context and incorporate more strength of various parsing techniques. However, higher-order constituent parsing inevitably leads to a high computational complexity. We intend to deal with the efficiency problem of our model with some advanced parallel computing technologies in our future works.

[2] ftp://ftp.cs.brown.edu/pub/nlparser/

[3] http://nlp.stanford.edu/software/lex-parser.shtml

[4] http://nlp.cs.nyu.edu/evalb/

References

E. Black, S. Abney, D. Flickenger, 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 DARPA Speech and Natural Language Workshop*, pages 306–311.

Rens Bod. 2003. An efficient implementation of a new DOP model. In *EACL 2003*, pages 19–26.

David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *EMNLP 2008*, pages 877–886.

Xavier Carreras, Michael Collins, and Terry Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *CoNLL 2008*, pages 9–16.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *EMNLP-CoNLL 2007*, pages 957–961.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *ACL 2005*, pages 173–180.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *NAACL 2000*, pages 132–139.

Xiao Chen and Chunyu Kit. 2011. Improving part-of-speech tagging for context-free parsing. In *IJCNLP 2011*, pages 1260–1268.

Xiao Chen. 2012. *Discriminative Constituent Parsing with Localized Features*. Ph.D. thesis, City University of Hong Kong.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *ACL 2002*, pages 263–270.

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *ICML 2000*, pages 175–182.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *EMNLP 2002*, pages 1–8.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996*, pages 340–345.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *ACL-HLT 2008*, pages 959–967.

Victoria Fossum and Kevin Knight. 2009. Combining constituent parsers. In *NAACL-HLT 2009*, pages 253–256.

James Henderson. 2004. Discriminative training of a neural network statistical parser. In *ACL 2004*, pages 95–102.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

Zhongqiang Huang, Mary Harper, and Slav Petrov. 2010. Self-training with products of latent variable grammars. In *EMNLP 2010*, pages 12–22.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *ACL-HLT 2008*, pages 586–594.

Mark Johnson. 2001. Joint and conditional estimation of tagging and parsing models. In *ACL 2001*, pages 322–329.

J. Kiefer. 1953. Sequential minimax search for a maximum. *Proceedings of the American Mathematical Society*, 4:502–506.

Terry Koo and Michael Collins. 2010. Efficient third-order dependency parsers. In *ACL 2010*, pages 1–11.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *EACL 2006*, pages 81–88.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *ACL 2005*, pages 91–98.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *EMNLP-HLT 2005*, pages 523–530.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *NAACL-HLT 2007*, pages 404–411.

Slav Petrov and Dan Klein. 2008a. Discriminative log-linear grammars with latent variables. In *NIPS 20*, pages 1–8.

Slav Petrov and Dan Klein. 2008b. Sparse multi-scale grammars for discriminative latent variable parsing. In *EMNLP 2008*, pages 867–876.

Slav Petrov. 2010. Products of random latent variable grammars. In *NAACL-HLT 2010*, pages 19–27.

M. J. D. Powell. 1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 7(2):155–162.

Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *NAACL-HLT 2006*, pages 129–132.

Ben Taskar, Dan Klein, Mike Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *EMNLP 2004*, pages 1–8.

Hui Zhang, Min Zhang, Chew Lim Tan, and Haizhou Li. 2009. K-best combination of syntactic parsers. In *EMNLP 2009*, pages 1552–1560.

Joint Evaluation of Morphological Segmentation and Syntactic Parsing

Reut Tsarfaty Joakim Nivre Evelina Andersson
Box 635, 751 26, Uppsala University, Uppsala, Sweden
tsarfaty@stp.lingfil.uu.se, {*joakim.nivre, evelina.andersson*}*@lingfil.uu.se*

Abstract

We present novel metrics for parse evaluation in joint segmentation and parsing scenarios where the gold sequence of terminals is not known in advance. The protocol uses distance-based metrics defined for the space of trees over lattices. Our metrics allow us to precisely quantify the performance gap between non-realistic parsing scenarios (assuming gold segmented and tagged input) and realistic ones (not assuming gold segmentation and tags). Our evaluation of segmentation and parsing for Modern Hebrew sheds new light on the performance of the best parsing systems to date in the different scenarios.

1 Introduction

A parser takes a sentence in natural language as input and returns a syntactic parse tree representing the sentence's human-perceived interpretation. Current state-of-the-art parsers assume that the space-delimited words in the input are the basic units of syntactic analysis. Standard evaluation procedures and metrics (Black et al., 1991; Buchholz and Marsi, 2006) accordingly assume that the yield of the parse tree is known in advance. This assumption breaks down when parsing morphologically rich languages (Tsarfaty et al., 2010), where every space-delimited word may be effectively composed of multiple morphemes, each of which having a distinct role in the syntactic parse tree. In order to parse such input the text needs to undergo *morphological segmentation*, that is, identifying the morphological segments of each word and assigning the corresponding part-of-speech (PoS) tags to them.

Morphologically complex words may be highly ambiguous and in order to segment them correctly their analysis has to be disambiguated. The multiple morphological analyses of input words may be represented via a lattice that encodes the different segmentation possibilities of the entire word sequence. One can either select a segmentation path prior to parsing, or, as has been recently argued, one can let the parser pick a segmentation jointly with decoding (Tsarfaty, 2006; Cohen and Smith, 2007; Goldberg and Tsarfaty, 2008; Green and Manning, 2010). If the selected segmentation is different from the gold segmentation, the gold and parse trees are rendered incomparable and standard evaluation metrics break down. Evaluation scenarios restricted to gold input are often used to bypass this problem, but, as shall be seen shortly, they present an overly optimistic upper-bound on parser performance.

This paper presents a full treatment of evaluation in different parsing scenarios, using distance-based measures defined for trees over a shared common denominator defined in terms of a lattice structure. We demonstrate the informativeness of our metrics by evaluating joint segmentation and parsing performance for the Semitic language Modern Hebrew, using the best performing systems, both constituency-based and dependency-based (Tsarfaty, 2010; Goldberg, 2011a). Our experiments demonstrate that, for all parsers, significant performance gaps between realistic and non-realistic scenarios crucially depend on the kind of information initially provided to the parser. The tool and metrics that we provide are completely general and can straightforwardly apply to other languages, treebanks and different tasks.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 6–10,
Jeju, Republic of Korea, 8-14 July 2012.

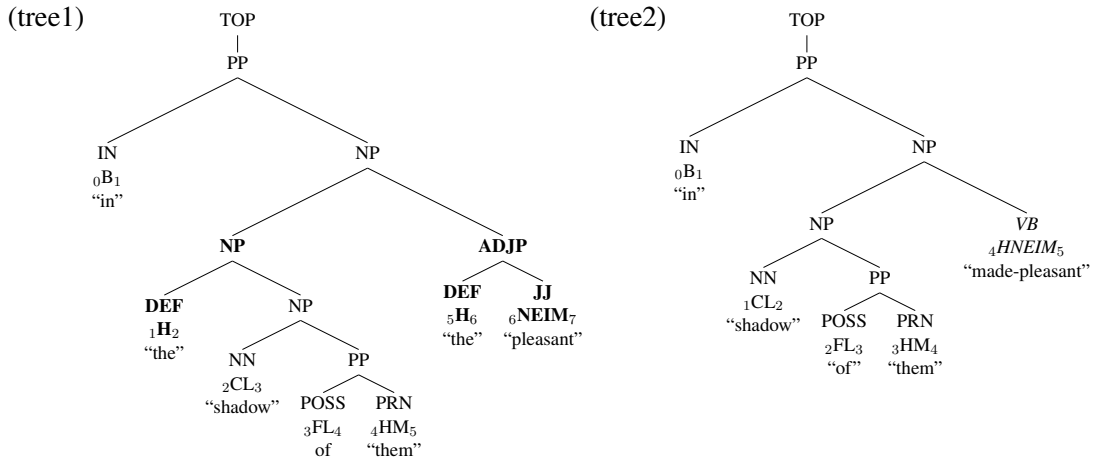


Figure 1: A correct tree (tree1) and an incorrect tree (tree2) for "BCLM HNEIM", indexed by terminal boundaries. Erroneous nodes in the parse hypothesis are marked in *italics*. Missing nodes from the hypothesis are marked in **bold**.

2 The Challenge: Evaluation for MRLs

In morphologically rich languages (MRLs) substantial information about the grammatical relations between entities is expressed at word level using inflectional affixes. In particular, in MRLs such as Hebrew, Arabic, Turkish or Maltese, elements such as determiners, definite articles and conjunction markers appear as affixes that are appended to an open-class word. Take, for example the Hebrew word-token BCLM,[1] which means "in their shadow". This word corresponds to five distinctly tagged elements: B ("in"/IN), H ("the"/DEF), CL ("shadow"/NN), FL ("of"/POSS), HM ("they"/PRN). Note that morphological segmentation is not the inverse of concatenation. For instance, the overt definite article H and the possessor FL show up only in the analysis.

The correct parse for the Hebrew phrase "BCLM HNEIM" is shown in Figure 1 (tree1), and it presupposes that these segments can be identified and assigned the correct PoS tags. However, morphological segmentation is non-trivial due to massive word-level ambiguity. The word BCLM, for instance, can be segmented into the noun BCL ("onion") and M (a genitive suffix, "of them"), or into the prefix B ("in") followed by the noun CLM ("image").[2] The multitude of morphological analyses may be encoded in a lattice structure, as illustrated in Figure 2.

[1] We use the Hebrew transliteration in Sima'an et al. (2001).

[2] The complete set of analyses for this word is provided in Goldberg and Tsarfaty (2008). Examples for similar phenomena in Arabic may be found in Green and Manning (2010).

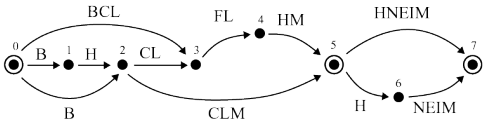


Figure 2: The morphological segmentation possibilities of BCLM HNEIM. Double-circles are word boundaries.

In practice, a statistical component is required to decide on the correct morphological segmentation, that is, to pick out the correct path through the lattice. This may be done based on linear local context (Adler and Elhadad, 2006; Shacham and Wintner, 2007; Bar-haim et al., 2008; Habash and Rambow, 2005), or jointly with parsing (Tsarfaty, 2006; Goldberg and Tsarfaty, 2008; Green and Manning, 2010). Either way, an incorrect morphological segmentation hypothesis introduces errors into the parse hypothesis, ultimately providing a parse tree which spans a different yield than the gold terminals. In such cases, existing evaluation metrics break down.

To understand why, consider the trees in Figure 1. Metrics like PARSEVAL (Black et al., 1991) calculate the harmonic means of precision and recall on labeled spans $\langle i, label, j \rangle$ where i, j are terminal boundaries. Now, the NP dominating "shadow of them" has been identified and labeled correctly in tree2, but in tree1 it spans $\langle 2, \text{NP}, 5 \rangle$ and in tree2 it spans $\langle 1, \text{NP}, 4 \rangle$. This node will then be counted as an error for tree2, along with its dominated and dominating structure, and PARSEVAL will score 0.

A generalized version of PARSEVAL which considers i, j character-based indices instead of terminal boundaries (Tsarfaty, 2006) will fail here too, since the missing overt definite article H will cause similar misalignments. Metrics for dependency-based evaluation such as ATTACHMENT SCORES (Buchholz and Marsi, 2006) suffer from similar problems, since they assume that both trees have the same nodes — an assumption that breaks down in the case of incorrect morphological segmentation.

Although great advances have been made in parsing MRLs in recent years, this evaluation challenge remained unsolved.[3] In this paper we present a solution to this challenge by extending TEDEVAL (Tsarfaty et al., 2011) for handling trees over lattices.

3 The Proposal: Distance-Based Metrics

Input and Output Spaces We view the joint task as a structured prediction function $h : \mathcal{X} \rightarrow \mathcal{Y}$ from input space $\mathcal{X}$ onto output space $\mathcal{Y}$. Each element $x \in \mathcal{X}$ is a sequence $x = w_1, \ldots, w_n$ of space-delimited words from a set $\mathcal{W}$. We assume a lexicon LEX, distinct from $\mathcal{W}$, containing pairs of segments drawn from a set $\mathcal{T}$ of terminals and PoS categories drawn from a set $\mathcal{N}$ of nonterminals.

$$\text{LEX} = \{\langle s, p\rangle | s \in \mathcal{T}, p \in \mathcal{N}\}$$

Each word w_i in the input may admit multiple morphological analyses, constrained by a language-specific morphological analyzer MA. The morphological analysis of an input word $\text{MA}(w_i)$ can be represented as a lattice L_i in which every arc corresponds to a lexicon entry $\langle s, p\rangle$. The morphological analysis of an input sentence x is then a lattice L obtained through the concatenation of the lattices $L_1, \ldots, L_n$ where $\text{MA}(w_1) = L_1, \ldots, \text{MA}(w_n) = L_n$. Now, let $x = w_1, \ldots, w_n$ be a sentence with a morphological analysis lattice $\text{MA}(x) = L$. We define the output space $\mathcal{Y}_{\text{MA}(x)=L}$ for h (abbreviated $\mathcal{Y}_L$), as the set of linearly-ordered labeled trees such that the yield of LEX entries $\langle s_1, p_1\rangle, \ldots, \langle s_k, p_k\rangle$ in each tree (where $s_i \in \mathcal{T}$ and $p_i \in \mathcal{N}$, and possibly $k \neq n$) corresponds to a path through the lattice L.

[3] A tool that could potentially apply here is SParseval (Roark et al., 2006). But since it does not respect word-boundaries, it fails to apply to such lattices. Cohen and Smith (2007) aimed to fix this, but in their implementation syntactic nodes internal to word boundaries may be lost without scoring.

Edit Scripts and Edit Costs We assume a set $\mathcal{A}$=$\{\text{ADD}(c, i, j), \text{DEL}(c, i, j), \text{ADD}(\langle s, p\rangle, i, j), \text{DEL}(\langle s, p\rangle, i, j)\}$ of edit operations which can add or delete a labeled node $c \in \mathcal{N}$ or an entry $\langle s, p\rangle \in$ LEX which spans the states i, j in the lattice L. The operations in $\mathcal{A}$ are properly constrained by the lattice, that is, we can only add and delete lexemes that belong to LEX, and we can only add and delete them where they can occur in the lattice. We assume a function $\text{C}(a) = 1$ assigning a unit cost to every operation $a \in \mathcal{A}$, and define the cost of a sequence $\langle a_1, \ldots, a_m\rangle$ as the sum of the costs of all operations in the sequence $\text{C}(\langle a_1, ..., a_m\rangle) = \sum_{i=1}^{m} \text{C}(a_i)$. An *edit script* $\text{ES}(y_1, y_2) = \langle a_1, \ldots, a_m\rangle$ is a sequence of operations that turns y_1 into y_2. The *tree-edit distance* is the minimum cost of any edit script that turns y_1 into y_2 (Bille, 2005).

$$\text{TED}(y_1, y_2) = \min_{\text{ES}(y_1, y_2)} \text{C}(\text{ES}(y_1, y_2))$$

Distance-Based Metrics The error of a predicted structure p with respect to a gold structure g is now taken to be the TED cost, and we can turn it into a score by normalizing it and subtracting from a unity:

$$\text{TEDEVAL}(p, g) = 1 - \frac{\text{TED}(p, g)}{|p| + |g| - 2}$$

The term $|p| + |g| - 2$ is a normalization factor defined in terms of the worst-case scenario, in which the parser has only made incorrect decisions. We would need to delete all lexemes and nodes in p and add all the lexemes and nodes of g, except for roots.

An Example Both trees in Figure 1 are contained in $\mathcal{Y}_L$ for the lattice L in Figure 2. If we replace terminal boundaries with lattice indices from Figure 2, we need 6 edit operations to turn tree2 into tree1 (deleting the nodes in *italic*, adding the nodes in **bold**) and the evaluation score will be $\text{TEDEVAL}(\text{tree2,tree1}) = 1 - \frac{6}{14+10-2} = 0.7273$.

4 Experiments

We aim to evaluate state-of-the-art parsing architectures on the morphosyntactic disambiguation of Hebrew texts in three different parsing scenarios: *(i) Gold:* assuming gold segmentation and PoS-tags, *(ii) Predicted:* assuming only gold segmentation, and *(iii) Raw:* assuming unanalyzed input text.

		SEGEVAL	PARSEVAL	TEDEVAL
Gold	PS	U: 100.00 L: 100.00	L: 88.75	U: 94.35 L: 93.39
Predicted	PS	U: 100.00 L: 90.85	L: 82.30	U: 92.92 L: 86:26
Raw	PS	U: 96.42 L: 84.54	N/A	U: 88.47 L: 80.67
Gold	RR	U: 100.00 L: 100.00	L: 83.93	U: 94.34 L: 92.45
Predicted	RR	U: 100.00 L: 91.69	L: 78.93	U: 92.82 L: 85.83
Raw	RR	U: 96.03 L: 86.10	N/A	U: 87.96 L: 79.46

Table 1: Phrase-Structure based results for the Berkeley Parser trained on bare-bone trees (PS) and relational-realizational trees (RR). We parse all sentences in the dev set. RR extra decoration is removed prior to evaluation.

		SEGEVAL	ATTSCORES	TEDEVAL
Gold	MP	100.00	U: 83.59	U: 91.76
Predicted	MP	100.00	U: 82.00	U: 91.20
Raw	MP	95.07	N/A	U: 87.03
Gold	EF	100.00	U: 84.68	U: 92.25
Predicted	EF	100.00	U: 83.97	U: 92:02
Raw	EF	95.07	N/A	U: 87.75

Table 2: Dependency parsing results by MaltParser (MP) and EasyFirst (EF), trained on the treebank converted into unlabeled dependencies, and parsing the entire dev-set.

For constituency-based parsing we use two models trained by the Berkeley parser (Petrov et al., 2006) one on phrase-structure (PS) trees and one on relational-realizational (RR) trees (Tsarfaty and Sima'an, 2008). In the *raw* scenario we let a lattice-based parser choose its own segmentation and tags (Goldberg, 2011b). For dependency parsing we use MaltParser (Nivre et al., 2007b) optimized for Hebrew by Ballesteros and Nivre (2012), and the EasyFirst parser of Goldberg and Elhadad (2010) with the features therein. Since these parsers cannot choose their own tags, automatically predicted segments and tags are provided by Adler and Elhadad (2006).

We use the standard split of the Hebrew treebank (Sima'an et al., 2001) and its conversion into unlabeled dependencies (Goldberg, 2011a). We use PARSEVAL for evaluating phrase-structure trees, ATTACHSCORES for evaluating dependency trees, and TEDEVAL for evaluating all trees in all scenarios. We implement SEGEVAL for evaluating segmentation based on our TEDEVAL implementation, replacing the tree distance and size with string terms.

Table 1 shows the constituency-based parsing results for all scenarios. All of our results confirm that gold information leads to much higher scores. TEDEVAL allows us to precisely quantify the drop in accuracy from *gold* to *predicted* (as in PARSEVAL) and than from *predicted* to *raw* on a single scale. TEDEVAL further allows us to scrutinize the contribution of different sorts of information. Unlabeled TEDEVAL shows a greater drop when moving from *predicted* to *raw* than from *gold* to *predicted*, and for labeled TEDEVAL it is the other way round. This demonstrates the great importance of gold tags which provide morphologically disambiguated information for identifying phrase content.

Table 2 shows that dependency parsing results confirm the same trends, but we see a much smaller drop when moving from *gold* to *predicted*. This is due to the fact that we train the parsers for *predicted* on a treebank containing *predicted* tags. There is however a great drop when moving from *predicted* to *raw*, which confirms that evaluation benchmarks on gold input as in Nivre et al. (2007a) do not provide a realistic indication of parser performance.

For all tables, TEDEVAL results are on a similar scale. However, results are not yet comparable across parsers. RR trees are flatter than bare-bone PS trees. PS and DEP trees have different label sets. Cross-framework evaluation may be conducted by combining this metric with the cross-framework protocol of Tsarfaty et al. (2012).

5 Conclusion

We presented distance-based metrics defined for trees over lattices and applied them to evaluating parsers on joint morphological and syntactic disambiguation. Our contribution is both technical, providing an evaluation tool that can be straightforwardly applied for parsing scenarios involving trees over lattices,[4] and methodological, suggesting to evaluate parsers in all possible scenarios in order to get a realistic indication of parser performance.

Acknowledgements

We thank Shay Cohen, Yoav Goldberg and Spence Green for discussion of this challenge. This work was supported by the Swedish Science Council.

[4]The tool can be downloaded `http://stp.ling.uu.se/~tsarfaty/unipar/index.html`

References

Meni Adler and Michael Elhadad. 2006. An unsupervised morpheme-based HMM for Hebrew morphological disambiguation. In *Proceedings of COLING-ACL*.

Miguel Ballesteros and Joakim Nivre. 2012. MaltOptimizer: A system for MaltParser optimization. Istanbul.

Roy Bar-haim, Khalil Sima'an, and Yoad Winter. 2008. Part-of-speech tagging of Modern Hebrew text. *Natural Language Engineering*, 14(2):223–251.

Philip Bille. 2005. A survey on tree-edit distance and related. problems. *Theoretical Computer Science*, 337:217–239.

Ezra Black, Steven P. Abney, D. Flickenger, Claudia Gdaniec, Ralph Grishman, P. Harrison, Donald Hindle, Robert Ingria, Frederick Jelinek, Judith L. Klavans, Mark Liberman, Mitchell P. Marcus, Salim Roukos, Beatrice Santorini, and Tomek Strzalkowski. 1991. A procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the DARPA Workshop on Speech and Natural Language*.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL-X*, pages 149–164.

Shay B. Cohen and Noah A. Smith. 2007. Joint morphological and syntactic disambiguation. In *Proceedings of EMNLP-CoNLL*, pages 208–217.

Yoav Goldberg and Michael Elhadad. 2010. Easy-first dependency parsing of Modern Hebrew. In *Proceedings of NAACL/HLT workshop on Statistical Parsing of Morphologically Rich Languages*.

Yoav Goldberg and Reut Tsarfaty. 2008. A single framework for joint morphological segmentation and syntactic parsing. In *Proceedings of ACL*.

Yoav Goldberg. 2011a. *Automatic Syntactic Processing of Modern Hebrew*. Ph.D. thesis, Ben-Gurion University of the Negev.

Yoav Goldberg. 2011b. Joint morphological segmentation and syntactic parsing using a PCFGLA lattice parser. In *Proceedings of ACL*.

Spence Green and Christopher D. Manning. 2010. Better Arabic parsing: Baselines, evaluations, and analysis. In *Proceedings of COLING*.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of ACL*.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007a. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932.

Joakim Nivre, Jens Nilsson, Johan Hall, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007b. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(1):1–41.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of ACL*.

Brian Roark, Mary Harper, Eugene Charniak, Bonnie Dorr C, Mark Johnson D, Jeremy G. Kahn E, Yang Liu F, Mari Ostendorf E, John Hale H, Anna Krasnyanskaya I, Matthew Lease D, Izhak Shafran J, Matthew Snover C, Robin Stewart K, and Lisa Yung J. 2006. Sparseval: Evaluation metrics for parsing speech. In *Proceesings of LREC*.

Danny Shacham and Shuly Wintner. 2007. Morphological disambiguation of Hebrew: A case study in classifier combination. In *Proceedings of the 2007 Joint Conference of EMNLP-CoNLL*, pages pages 439–447.

Khalil Sima'an, Alon Itai, Yoad Winter, Alon Altman, and Noa Nativ. 2001. Building a Tree-Bank for Modern Hebrew Text. In *Traitement Automatique des Langues*.

Reut Tsarfaty and Khalil Sima'an. 2008. Relational-Realizational parsing. In *Proceedings of CoLing*.

Reut Tsarfaty, Djame Seddah, Yoav Goldberg, Sandra Kuebler, Marie Candito, Jennifer Foster, Yannick Versley, Ines Rehbein, and Lamia Tounsi. 2010. Statistical parsing for morphologically rich language (SPMRL): What, how and whither. In *Proceedings of the first workshop on Statistical Parsing of Morphologically Rich Languages (SPMRL) at NA-ACL*.

Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2011. Evaluating dependency parsing: Robust and heuristics-free cross-framework evaluation. In *Proceedings of EMNLP*.

Reut Tsarfaty, Joakim Nivre, and Evelina Andersson. 2012. Cross-framework evaluation for statistical parsing. In *Proceedings of EACL*.

Reut Tsarfaty. 2006. Integrated morphological and syntactic disambiguation for Modern Hebrew. In *Proceeding of ACL-SRW*.

Reut Tsarfaty. 2010. *Relational-Realizational Parsing*. Ph.D. thesis, University of Amsterdam.

A Comparison of Chinese Parsers for Stanford Dependencies

Wanxiang Che[†]
car@ir.hit.edu.cn

Valentin I. Spitkovsky[‡]
vals@stanford.edu

Ting Liu[†]
tliu@ir.hit.edu.cn

[†]School of Computer Science and Technology
Harbin Institute of Technology
Harbin, China, 150001

[‡]Computer Science Department
Stanford University
Stanford, CA, 94305

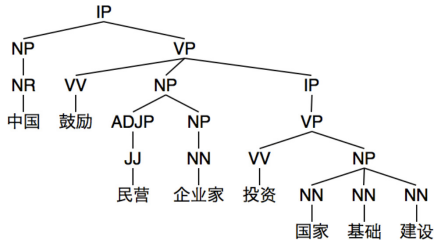
Abstract

Stanford dependencies are widely used in natural language processing as a semantically-oriented representation, commonly generated either by (i) converting the output of a constituent parser, or (ii) predicting dependencies directly. Previous comparisons of the two approaches for English suggest that starting from constituents yields higher accuracies. In this paper, we re-evaluate both methods for Chinese, using more accurate dependency parsers than in previous work. Our comparison of performance and efficiency across seven popular open source parsers (four constituent and three dependency) shows, by contrast, that recent higher-order graph-based techniques can be more accurate, though somewhat slower, than constituent parsers. We demonstrate also that n-way jackknifing is a useful technique for producing automatic (rather than gold) part-of-speech tags to train Chinese dependency parsers. Finally, we analyze the relations produced by both kinds of parsing and suggest which specific parsers to use in practice.

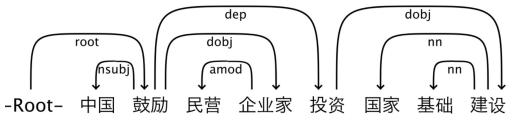
1 Introduction

Stanford dependencies (de Marneffe and Manning, 2008) provide a simple description of relations between pairs of words in a sentence. This semantically-oriented representation is intuitive and easy to apply, requiring little linguistic expertise. Consequently, Stanford dependencies are widely used: in biomedical text mining (Kim et al., 2009), as well as in textual entailment (Androutsopoulos and Malakasiotis, 2010), information extraction (Wu and Weld, 2010; Banko et al., 2007) and sentiment analysis (Meena and Prabhakar, 2007).

In addition to English, there is a Chinese version of Stanford dependencies (Chang et al., 2009),



(a) A constituent parse tree.



(b) Stanford dependencies.

Figure 1: A sample Chinese constituent parse tree and its corresponding Stanford dependencies for the sentence *China* (中国) *encourages* (鼓励) *private* (民营) *entrepreneurs* (企业家) *to invest* (投资) *in national* (国家) *infrastructure* (基础) *construction* (建设).

which is also useful for many applications, such as Chinese sentiment analysis (Wu et al., 2011; Wu et al., 2009; Zhuang et al., 2006) and relation extraction (Huang et al., 2008). Figure 1 shows a sample constituent parse tree and the corresponding Stanford dependencies for a sentence in Chinese. Although there are several variants of Stanford dependencies for English,[1] so far only a basic version (i.e, dependency tree structures) is available for Chinese.

Stanford dependencies were originally obtained from constituent trees, using rules (de Marneffe et al., 2006). But as dependency parsing technologies mature (Kübler et al., 2009), they offer increasingly attractive alternatives that eliminate the need for an intermediate representation. Cer et al. (2010) reported that Stanford's implementation (Klein and Manning, 2003) underperforms other constituent

[1] nlp.stanford.edu/software/dependencies_manual.pdf

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 11–16,
Jeju, Republic of Korea, 8-14 July 2012.

Type	*Parser*	*Version*	*Algorithm*	URL
Constituent	Berkeley	1.1	PCFG	code.google.com/p/berkeleyparser
	Bikel	1.2	PCFG	www.cis.upenn.edu/~dbikel/download.html
	Charniak	Nov. 2009	PCFG	www.cog.brown.edu/~mj/Software.htm
	Stanford	2.0	Factored	nlp.stanford.edu/software/lex-parser.shtml
Dependency	MaltParser	1.6.1	Arc-Eager	maltparser.org
	Mate	2.0	2nd-order MST	code.google.com/p/mate-tools
	MSTParser	0.5	MST	sourceforge.net/projects/mstparser

Table 1: Basic information for the seven parsers included in our experiments.

parsers, for English, on both accuracy and speed. Their thorough investigation also showed that constituent parsers systematically outperform parsing directly to Stanford dependencies. Nevertheless, relative standings could have changed in recent years: dependency parsers are now significantly more accurate, thanks to advances like the high-order maximum spanning tree (MST) model (Koo and Collins, 2010) for graph-based dependency parsing (McDonald and Pereira, 2006). Therefore, we deemed it important to re-evaluate the performance of constituent and dependency parsers. But the main purpose of our work is to apply the more sophisticated dependency parsing algorithms specifically to Chinese.

Number of \in	*Train*	*Dev*	*Test*	*Total*
files	2,083	160	205	2,448
sentences	46,572	2,079	2,796	51,447
tokens	1,039,942	59,955	81,578	1,181,475

Table 2: Statistics for Chinese TreeBank (CTB) 7.0 data.

2 Methodology

We compared seven popular open source constituent and dependency parsers, focusing on both accuracy and parsing speed. We hope that our analysis will help end-users select a suitable method for parsing to Stanford dependencies in their own applications.

2.1 Parsers

We considered four constituent parsers. They are: Berkeley (Petrov et al., 2006), Bikel (2004), Charniak (2000) and Stanford (Klein and Manning, 2003) `chineseFactored`, which is also the default used by Stanford dependencies. The three dependency parsers are: MaltParser (Nivre et al., 2006), Mate (Bohnet, 2010)[2] and MSTParser (McDonald and Pereira, 2006). Table 1 has more information.

[2]A second-order MST parser (with the speed optimization).

2.2 Corpus

We used the latest Chinese TreeBank (CTB) 7.0 in all experiments.[3] CTB 7.0 is larger and has more sources (e.g., web text), compared to previous versions. We split the data into train/development/test sets (see Table 2), with gold word segmentation, following the guidelines suggested in documentation.

2.3 Settings

Every parser was run with its own default options. However, since the default classifier used by MaltParser is *libsvm* (Chang and Lin, 2011) with a polynomial kernel, it may be too slow for training models on all of CTB 7.0 training data in acceptable time. Therefore, we also tested this particular parser with the faster *liblinear* (Fan et al., 2008) classifier. All experiments were performed on a machine with Intel's Xeon E5620 2.40GHz CPU and 24GB RAM.

2.4 Features

Unlike constituent parsers, dependency models require exogenous part-of-speech (POS) tags, both in training and in inference. We used the Stanford tagger (Toutanova et al., 2003) v3.1, with the MEMM model,[4] in combination with 10-way jackknifing.[5]

Word lemmas — which are generalizations of words — are another feature known to be useful for dependency parsing. Here we lemmatized each Chinese word down to its last character, since — in contrast to English — a Chinese word's suffix often carries that word's core sense (Tseng et al., 2005). For example, *bicycle* (自行车), *car* (汽车) and *train* (火车) are all various kinds of *vehicle* (车).

[3]www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2010T07

[4]nlp.stanford.edu/software/tagger.shtml

[5]Training sentences in each fold were tagged using a model based on the other nine folds; development and test sentences were tagged using a model based on all ten of the training folds.

		Dev		Test		
Type	*Parser*	UAS	LAS	UAS	LAS	*Parsing Time*
Constituent	Berkeley	*82.0*	*77.0*	*82.9*	*77.8*	*45:56*
	Bikel	79.4	74.1	80.0	74.3	*6,861:31*
	Charniak	77.8	71.7	78.3	72.3	128:04
	Stanford	*76.9*	*71.2*	*77.3*	*71.4*	330:50
Dependency	MaltParser (*liblinear*)	*76.0*	*71.2*	*76.3*	*71.2*	**0:11**
	MaltParser (*libsvm*)	77.3	72.7	78.0	73.1	*556:51*
	Mate (2nd-order)	**82.8**	**78.2**	**83.1**	**78.1**	87:19
	MSTParser (1st-order)	78.8	73.4	78.9	73.1	12:17

Table 3: Performance and efficiency for all parsers on CTB data: unlabeled and labeled attachment scores (UAS/LAS) are for both development and test data sets; parsing times (minutes:seconds) are for the test data only and exclude generation of basic Stanford dependencies (for constituent parsers) and part-of-speech tagging (for dependency parsers).

3 Results

Table 3 tabulates efficiency and performance for all parsers; UAS and LAS are unlabeled and labeled attachment scores, respectively — the standard criteria for evaluating dependencies. They can be computed via a CoNLL-X shared task dependency parsing evaluation tool (without scoring punctuation).[6]

3.1 Chinese

Mate scored highest, and Berkeley was the most accurate of constituent parsers, slightly behind Mate, using half of the time. MaltParser (*liblinear*) was by far the most efficient but also the least performant; it scored higher with *libsvm* but took much more time.

The 1st-order MSTParser was more accurate than MaltParser (*libsvm*) — a result that differs from that of Cer et al. (2010) for English (see §3.2). The Stanford parser (the default for Stanford dependencies) was only slightly more accurate than MaltParser (*liblinear*). Bikel's parser was too slow to be used in practice; and Charniak's parser — which performs best for English — did not work well for Chinese.

3.2 English

Our replication of Cer et al.'s (2010, Table 1) evaluation revealed a bug: MSTParser normalized all numbers to a <num> symbol, which decreased its scores in the evaluation tool used with Stanford dependencies. After fixing this glitch, MSTParser's performance improved from 78.8 (reported) to 82.5%, thus making it more accurate than MaltParser (81.1%) and hence the better dependency parser for English, consistent with our results for Chinese (see Table 3).

Our finding does *not* contradict the main qualitative result of Cer et al. (2010), however, since the constituent parser of Charniak and Johnson (2005) still scores substantially higher (89.1%), for English, compared to all dependency parsers.[7] In a separate experiment (parsing web data),[8] we found Mate to be less accurate than Charniak-Johnson — and improvement from jackknifing smaller — on English.

4 Analysis

To further compare the constituent and dependency approaches to generating Stanford dependencies, we focused on Mate and Berkeley parsers — the best of each type. Overall, the difference between their accuracies is not statistically significant ($p > 0.05$).[9]

Table 4 highlights performance (F_1 scores) for the most frequent relation labels. Mate does better on most relations, noun compound modifiers (*nn*) and adjectival modifiers (*amod*) in particular; and the Berkeley parser is better at *root* and *dep*.[10] Mate seems to excel at short-distance dependencies, possibly because it uses more local features (even with a second-order model) than the Berkeley parser, whose PCFG can capture longer-distance rules.

Since POS-tags are especially informative of Chinese dependencies (Li et al., 2011), we harmonized training and test data, using 10-way jackknifing (see §2.4). This method is more robust than training a

[6] ilk.uvt.nl/conll/software/eval.pl

[7] One (small) factor contributing to the difference between the two languages is that in the Chinese setup we stop with basic Stanford dependencies — there is no penalty for further conversion; another is not using discriminative reranking for Chinese.

[8] sites.google.com/site/sancl2012/home/shared-task

[9] For LAS, $p \approx 0.11$; and for UAS, $p \approx 0.25$, according to www.cis.upenn.edu/~dbikel/download/compare.pl

[10] An unmatched (default) relation (Chang et al., 2009, §3.1).

Relation	*Count*	Mate	Berkeley
nn	7,783	**91.3**	89.3
dep	4,651	69.4	**70.3**
nsubj	4,531	**87.1**	85.5
advmod	4,028	**94.3**	93.8
dobj	3,990	**86.0**	85.0
conj	2,159	**76.0**	75.8
prep	2,091	**94.3**	94.1
root	2,079	81.2	**82.3**
nummod	1,614	**97.4**	96.7
assmod	1,593	**86.3**	84.1
assm	1,590	**88.9**	87.2
pobj	1,532	**84.2**	82.9
amod	1,440	**85.6**	81.1
rcmod	1,433	**74.0**	70.6
cpm	1,371	**84.4**	83.2

Table 4: Performance (F_1 scores) for the fifteen most-frequent dependency relations in the CTB 7.0 development data set attained by both Mate and Berkeley parsers.

parser with gold tags because it improves consistency, particularly for Chinese, where tagging accuracies are lower than in English. On development data, Mate scored worse given gold tags (75.4 versus 78.2%).[11] Lemmatization offered additional useful cues for overcoming data sparseness (77.8 without, versus 78.2% with lemma features). Unsupervised word clusters could thus also help (Koo et al., 2008).

5 Discussion

Our results suggest that if accuracy is of primary concern, then Mate should be preferred;[12] however, Berkeley parser offers a trade-off between accuracy and speed. If neither parser satisfies the demands of a practical application (e.g., real-time processing or bulk-parsing the web), then MaltParser (*liblinear*) may be the only viable option. Fortunately, it comes with much headroom for improving accuracy, including a tunable margin parameter C for the classifier, richer feature sets (Zhang and Nivre, 2011) and ensemble models (Surdeanu and Manning, 2010).

Stanford dependencies are not the only popular dependency representation. We also considered the conversion scheme of the Penn2Malt tool,[13] used in a series of CoNLL shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007; Surdeanu et al., 2008; Hajič et al., 2009). However, this tool relies on function tag information from the CTB in determining dependency relations. Since these tags usually cannot be produced by constituent parsers, we could not, in turn, obtain CoNLL-style dependency trees from their output. This points to another advantage of dependency parsers: they need only the dependency tree corpus to train and can conveniently make use of native (unconverted) corpora, such as the Chinese Dependency Treebank (Liu et al., 2006).

Lastly, we must note that although the Berkeley parser is on par with Charniak's (2000) system for English (Cer et al., 2010, Table 1), its scores for Chinese are substantially higher. There may be subtle biases in Charniak's approach (e.g., the conditioning hierarchy used in smoothing) that could turn out to be language-specific. The Berkeley parser appears more general — without quite as many parameters or idiosyncratic design decisions — as evidenced by a recent application to French (Candito et al., 2010).

6 Conclusion

We compared seven popular open source parsers — four constituent and three dependency — for generating Stanford dependencies in Chinese. Mate, a high-order MST dependency parser, with lemmatization and jackknifed POS-tags, appears most accurate; but Berkeley's faster constituent parser, with jointly-inferred tags, is statistically no worse. This outcome is different from English, where constituent parsers systematically outperform direct methods.

Though Mate scored higher overall, Berkeley's parser was better at recovering longer-distance relations, suggesting that a combined approach could perhaps work better still (Rush et al., 2010, §4.2).

Acknowledgments

We thank Daniel Cer, for helping us replicate the English experimental setup and for suggesting that we explore jackknifing methods, and the anonymous reviewers, for valuable comments.

Supported in part by the National Natural Science Foundation of China (NSFC) via grant 61133012, the National "863" Major Project grant 2011AA01A207, and the National "863" Leading Technology Research Project grant 2012AA011102.

[11] Berkeley's performance suffered with jackknifed tags (76.5 versus 77.0%), possibly because it parses and tags better jointly.

[12] Although Mate's performance was not significantly better than Berkeley's in our setting, it has the potential to tap richer features and other advantages of dependency parsers (Nivre and McDonald, 2008) to further boost accuracy, which may be difficult in the generative framework of a typical constituent parser.

[13] `w3.msi.vxu.se/~nivre/research/Penn2Malt.html`

Second author gratefully acknowledges the continued help and support of his advisor, Dan Jurafsky, and of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program, under the 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 views of DARPA, AFRL, or the US government.

References

Ion Androutsopoulos and Prodromos Malakasiotis. 2010. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38(1):135–187, May.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2670–2676, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Daniel M. Bikel. 2004. A distributional analysis of a lexicalized statistical parsing model. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 182–189, Barcelona, Spain, July. Association for Computational Linguistics.

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China, August. Coling 2010 Organizing Committee.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, June. Association for Computational Linguistics.

Marie Candito, Joakim Nivre, Pascal Denis, and Enrique Henestroza Anguiano. 2010. Benchmarking of statistical dependency parsers for French. In *Coling 2010: Posters*, pages 108–116, Beijing, China, August. Coling 2010 Organizing Committee.

Daniel Cer, Marie-Catherine de Marneffe, Daniel Jurafsky, and Christopher D. Manning. 2010. Parsing to Stanford dependencies: Trade-offs between speed and accuracy. In *Proceedings of the 7th International Conference on Language Resources and Evaluation (LREC 2010)*.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, May.

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. Discriminative reordering with Chinese grammatical relations features. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation*, Boulder, Colorado, June.

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, June. Association for Computational Linguistics.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, NAACL 2000, pages 132–139, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, June.

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 *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 1–18, Boulder, Colorado, June. Association for Computational Linguistics.

Ruihong Huang, Le Sun, and Yuanyong Feng. 2008. Study of kernel-based methods for Chinese relation extraction. In *Proceedings of the 4th Asia information retrieval conference on Information retrieval technology*, AIRS'08, pages 598–604, Berlin, Heidelberg. Springer-Verlag.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, BioNLP '09, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. 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*, ACL '10, pages 1–11, Stroudsburg, PA, USA. 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.

Sandra Kübler, Ryan T. McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

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, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Ting Liu, Jinshan Ma, and Sheng Li. 2006. Building a dependency treebank for improving Chinese parser. *Journal of Chinese Language and Computing*, 16(4).

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of the 11th Conference of the European Chapter of the ACL (EACL 2006)*, pages 81–88.

Arun Meena and T. V. Prabhakar. 2007. Sentence level sentiment analysis in the presence of conjuncts using linguistic analysis. In *Proceedings of the 29th European conference on IR research*, ECIR'07, pages 573–580, Berlin, Heidelberg. Springer-Verlag.

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.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, pages 2216–2219.

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, Prague, Czech Republic, June. Association for Computational Linguistics.

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.

Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Cambridge, MA, October. Association for Computational Linguistics.

Mihai Surdeanu and Christopher D. Manning. 2010. Ensemble models for dependency parsing: cheap and good? In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 649–652, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England, August. Coling 2008 Organizing Committee.

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.

Huihsin Tseng, Daniel Jurafsky, and Christopher Manning. 2005. Morphological features help POS tagging of unknown words across language varieties. In *Proceedings of the fourth SIGHAN bakeoff*.

Fei Wu and Daniel S. Weld. 2010. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 118–127, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yuanbin Wu, Qi Zhang, Xuanjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1533–1541, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yuanbin Wu, Qi Zhang, Xuanjing Huang, and Lide Wu. 2011. Structural opinion mining for graph-based sentiment representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1332–1341, Stroudsburg, PA, USA. 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: short papers - Volume 2*, HLT '11, pages 188–193, Stroudsburg, PA, USA. Association for Computational Linguistics.

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *Proceedings of the 15th ACM international conference on Information and knowledge management*, CIKM '06, pages 43–50, New York, NY, USA. ACM.

A Feature-Rich Constituent Context Model for Grammar Induction

Dave Golland
University of California, Berkeley
dsg@cs.berkeley.edu

John DeNero
Google
denero@google.com

Jakob Uszkoreit
Google
uszkoreit@google.com

Abstract

We present LLCCM, a log-linear variant of the constituent context model (CCM) of grammar induction. LLCCM retains the simplicity of the original CCM but extends robustly to long sentences. On sentences of up to length 40, LLCCM outperforms CCM by 13.9% bracketing F1 and outperforms a right-branching baseline in regimes where CCM does not.

1 Introduction

Unsupervised grammar induction is a fundamental challenge of statistical natural language processing (Lari and Young, 1990; Pereira and Schabes, 1992; Carroll and Charniak, 1992). The constituent context model (CCM) for inducing constituency parses (Klein and Manning, 2002) was the first unsupervised approach to surpass a right-branching baseline. However, the CCM only effectively models short sentences. This paper shows that a simple reparameterization of the model, which ties together the probabilities of related events, allows the CCM to extend robustly to long sentences.

Much recent research has explored dependency grammar induction. For instance, the dependency model with valence (DMV) of Klein and Manning (2004) has been extended to utilize multilingual information (Berg-Kirkpatrick and Klein, 2010; Cohen et al., 2011), lexical information (Headden III et al., 2009), and linguistic universals (Naseem et al., 2010). Nevertheless, simplistic dependency models like the DMV do not contain information present in a constituency parse, such as the attachment order of object and subject to a verb.

Unsupervised constituency parsing is also an active research area. Several studies (Seginer, 2007; Reichart and Rappoport, 2010; Ponvert et al., 2011) have considered the problem of inducing parses over raw lexical items rather than part-of-speech (POS) tags. Additional advances have come from more complex models, such as combining CCM and DMV (Klein and Manning, 2004) and modeling large tree fragments (Bod, 2006).

The CCM scores each parse as a product of probabilities of span and context subsequences. It was originally evaluated only on unpunctuated sentences up to length 10 (Klein and Manning, 2002), which account for only 15% of the WSJ corpus; our experiments confirm the observation in (Klein, 2005) that performance degrades dramatically on longer sentences. This problem is unsurprising: CCM scores each constituent type by a single, isolated multinomial parameter.

Our work leverages the idea that sharing information between local probabilities in a structured unsupervised model can lead to substantial accuracy gains, previously demonstrated for dependency grammar induction (Cohen and Smith, 2009; Berg-Kirkpatrick et al., 2010). Our model, Log-Linear CCM (LLCCM), shares information between the probabilities of related constituents by expressing them as a log-linear combination of features trained using the gradient-based learning procedure of Berg-Kirkpatrick et al. (2010). In this way, the probability of generating a constituent is informed by related constituents.

Our model improves unsupervised constituency parsing of sentences longer than 10 words. On sentences of up to length 40 (96% of all sentences in the Penn Treebank), LLCCM outperforms CCM by 13.9% (unlabeled) bracketing F1 and, unlike CCM, outperforms a right-branching baseline on sentences longer than 15 words.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 17–22,
Jeju, Republic of Korea, 8-14 July 2012.

2 Model

The CCM is a generative model for the unsupervised induction of binary constituency parses over sequences of part-of-speech (POS) tags (Klein and Manning, 2002). Conditioned on the constituency or distituency of each span in the parse, CCM generates both the complete sequence of terminals it contains and the terminals in the surrounding context.

Formally, the CCM is a probabilistic model that jointly generates a sentence, s, and a bracketing, B, specifying whether each contiguous subsequence is a constituent or not, in which case the span is called a distituent. Each subsequence of POS tags, or SPAN, α, occurs in a CONTEXT, β, which is an ordered pair of preceding and following tags. A bracketing is a boolean matrix B, indicating which spans (i, j) are constituents ($\mathrm{B}_{ij} = true$) and which are distituents ($\mathrm{B}_{ij} = false$). A bracketing is considered legal if its constituents are nested and form a binary tree $T(\mathrm{B})$.

The joint distribution is given by:

$$\mathrm{P}(s, \mathrm{B}) = \mathrm{P}_{\mathrm{T}}(\mathrm{B}) \cdot \prod_{i,j \in T(\mathrm{B})} \mathrm{P}_{\mathrm{S}}(\alpha(i,j,s)|true)\, \mathrm{P}_{\mathrm{C}}(\beta(i,j,s)|true) \cdot \prod_{i,j \notin T(B)} \mathrm{P}_{\mathrm{S}}(\alpha(i,j,s)|false)\, \mathrm{P}_{\mathrm{C}}(\beta(i,j,s)|false)$$

The prior over unobserved bracketings $\mathrm{P}_{\mathrm{T}}(\mathrm{B})$ is fixed to be the uniform distribution over all legal bracketings. The other distributions, $\mathrm{P}_{\mathrm{S}}(\cdot)$ and $\mathrm{P}_{\mathrm{C}}(\cdot)$, are multinomials whose isolated parameters are estimated to maximize the likelihood of a set of observed sentences $\{s_n\}$ using EM (Dempster et al., 1977).[1]

2.1 The Log-Linear CCM

A fundamental limitation of the CCM is that it contains a single isolated parameter for every span. The number of different possible span types increases exponentially in span length, leading to data sparsity as the sentence length increases.

[1] As mentioned in (Klein and Manning, 2002), the CCM model is deficient because it assigns probability mass to yields and spans that cannot consistently combine to form a valid sentence. Our model does not address this issue, and hence it is similarly deficient.

The Log-Linear CCM (LLCCM) reparameterizes the distributions in the CCM using intuitive features to address the limitations of CCM while retaining its predictive power. The set of proposed features includes a BASIC feature for each parameter of the original CCM, enabling the LLCCM to retain the full expressive power of the CCM. In addition, LLCCM contains a set of coarse features that activate across distinct spans.

To introduce features into the CCM, we express each of its local conditional distributions as a multiclass logistic regression model. Each local distribution, $\mathrm{P}_t(y|x)$ for $t \in \{\text{SPAN}, \text{CONTEXT}\}$, conditions on label $x \in \{true, false\}$ and generates an event (span or context) y. We can define each local distribution in terms of a weight vector, $\mathbf{w}$, and feature vector, $\mathbf{f}_{xyt}$, using a log-linear model:

$$\mathrm{P}_t(y|x) = \frac{\exp \langle \mathbf{w}, \mathbf{f}_{xyt} \rangle}{\sum_{y'} \exp \langle \mathbf{w}, \mathbf{f}_{xy't} \rangle} \quad (1)$$

This technique for parameter transformation was shown to be effective in unsupervised models for part-of-speech induction, dependency grammar induction, word alignment, and word segmentation (Berg-Kirkpatrick et al., 2010). In our case, replacing multinomials via featurized models not only improves model accuracy, but also lets the model apply effectively to a new regime of long sentences.

2.2 Feature Templates

In the SPAN model, for each span $y = [\alpha_1, \ldots, \alpha_n]$ and label x, we use the following feature templates:

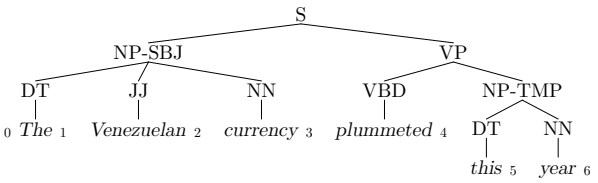
BASIC:	$\mathbb{I}[y = \cdot \wedge x = \cdot]$
BOUNDARY:	$\mathbb{I}[\alpha_1 = \cdot \wedge \alpha_n = \cdot \wedge x = \cdot]$
PREFIX:	$\mathbb{I}[\alpha_1 = \cdot \wedge x = \cdot]$
SUFFIX:	$\mathbb{I}[\alpha_n = \cdot \wedge x = \cdot]$

Just as the external CONTEXT is a signal of constituency, so too is the internal "context." For example, there are many distinct noun phrases with different spans that all begin with DT and end with NN; a fact expressed by the BOUNDARY feature (Table 1).

In the CONTEXT model, for each context $y = [\beta_1, \beta_2]$ and constituent/distituent decision x, we use the following feature templates:

BASIC:	$\mathbb{I}[y = \cdot \wedge x = \cdot]$
L-CONTEXT:	$\mathbb{I}[\beta_1 = \cdot \wedge x = \cdot]$
R-CONTEXT:	$\mathbb{I}[\beta_2 = \cdot \wedge x = \cdot]$

Consider the following example extracted from the WSJ:



Both spans (0, 3) and (4, 6) are constituents corresponding to noun phrases whose features are shown in Table 1:

	Feature Name	(0,3)	(4, 6)
span	BASIC-DT-JJ-NN:	1	0
	BASIC-DT-NN:	0	1
	BOUNDARY-DT-NN:	1	1
	PREFIX-DT:	1	1
	SUFFIX-NN:	1	1
context	BASIC-◇-VBD:	1	0
	BASIC-VBD-◇:	0	1
	L-CONTEXT-◇:	1	0
	L-CONTEXT-VBD:	0	1
	R-CONTEXT-VBD:	1	0
	R-CONTEXT-◇:	0	1

Table 1: Span and context features for constituent spans (0, 3) and (4, 6). The symbol ◇ indicates a sentence boundary.

Notice that although the BASIC span features are active for at most one span, the remaining features fire for both spans, effectively sharing information between the local probabilities of these events.

The coarser CONTEXT features factor the context pair into its components, which allow the LLCCM to more easily learn, for example, that a constituent is unlikely to immediately follow a determiner.

3 Training

In the EM algorithm for estimating CCM parameters, the E-Step computes posteriors over bracketings using the Inside-Outside algorithm. The M-Step chooses parameters that maximize the expected complete log likelihood of the data.

The weights, $\mathbf{w}$, of LLCCM are estimated to maximize the data log likelihood of the training sentences $\{s_n\}$, summing out all possible bracketings B for each sentence:

$$L(\mathbf{w}) = \sum_{s_n} \log \sum_{B} \mathrm{P}_{\mathbf{w}}(s_n, B)$$

We optimize this objective via L-BFGS (Liu and Nocedal, 1989), which requires us to compute the objective gradient. Berg-Kirkpatrick et al. (2010) showed that the data log likelihood gradient is equivalent to the gradient of the expected complete log likelihood (the objective maximized in the M-step of EM) at the point from which expectations are computed. This gradient can be computed in three steps.

First, we compute the local probabilities of the CCM, $\mathrm{P}_t(y|x)$, from the current $\mathbf{w}$ using Equation (1). We approximate the normalization over an exponential number of terms by only summing over spans that appeared in the training corpus.

Second, we compute posteriors over bracketings, $\mathrm{P}(i, j|s_n)$, just as in the E-step of CCM training,[2] in order to determine the expected counts:

$$e_{xy,\textsc{Span}} = \sum_{s_n} \sum_{ij} \mathbb{I}\left[\alpha(i, j, s_n) = y\right] \delta(x)$$

$$e_{xy,\textsc{Context}} = \sum_{s_n} \sum_{ij} \mathbb{I}\left[\beta(i, j, s_n) = y\right] \delta(x)$$

where $\delta(true) = \mathrm{P}(i, j|s_n)$, and $\delta(false) = 1 - \delta(true)$.

We summarize these expected count quantities as:

$$e_{xyt} = \begin{cases} e_{xy,\textsc{Span}} & \text{if } t = \textsc{Span} \\ e_{xy,\textsc{Context}} & \text{if } t = \textsc{Context} \end{cases}$$

Finally, we compute the gradient with respect to $\mathbf{w}$, expressed in terms of these expected counts and conditional probabilities:

$$\nabla L(\mathbf{w}) = \sum_{xyt} e_{xyt} \mathbf{f}_{xyt} - G(\mathbf{w})$$

$$G(\mathbf{w}) = \sum_{xt} \left(\sum_{y} e_{xyt} \right) \sum_{y'} \mathrm{P}_t(y|x) \mathbf{f}_{xy't}$$

Following (Klein and Manning, 2002), we initialize the model weights by optimizing against posterior probabilities fixed to the split-uniform distribution, which generates binary trees by randomly choosing a split point and recursing on each side of the split.[3]

[2] We follow the dynamic program presented in Appendix A.1 of (Klein, 2005).

[3] In Appendix B.2, Klein (2005) shows this posterior can be expressed in closed form. As in previous work, we start the initialization optimization with the zero vector, and terminate after 10 iterations to regularize against achieving a local maximum.

3.1 Efficiently Computing the Gradient

The following quantity appears in $G(\mathbf{w})$:

$$\gamma_t(x) = \sum_y e_{xyt}$$

Which expands as follows depending on t:

$$\gamma_{\text{SPAN}}(x) = \sum_y \sum_{s_n} \sum_{ij} \mathbb{I}\left[\alpha(i,j,s_n) = y\right] \delta(x)$$

$$\gamma_{\text{CONTEXT}}(x) = \sum_y \sum_{s_n} \sum_{ij} \mathbb{I}\left[\beta(i,j,s_n) = y\right] \delta(x)$$

In each of these expressions, the $\delta(x)$ term can be factored outside the sum over y. Each fixed (i,j) and s_n pair has exactly one span and context, hence the quantities $\sum_y \mathbb{I}\left[\alpha(i,j,s_n) = y\right]$ and $\sum_y \mathbb{I}\left[\beta(i,j,s_n) = y\right]$ are both equal to 1.

$$\gamma_t(x) = \sum_{s_n} \sum_{ij} \delta(x)$$

This expression further simplifies to a constant. The sum of the posterior probabilities, $\delta(true)$, over all positions is equal to the total number of constituents in the tree. Any binary tree over N terminals contains exactly $2N-1$ constituents and $\frac{1}{2}(N-2)(N-1)$ distituents.

$$\gamma_t(x) = \begin{cases} \sum_{s_n} (2|s_n| - 1) & \text{if } x = true \\ \frac{1}{2} \sum_{s_n} (|s_n| - 2)(|s_n| - 1) & \text{if } x = false \end{cases}$$

where $|s_n|$ denotes the length of sentence s_n.

Thus, $G(\mathbf{w})$ can be precomputed once for the entire dataset at each minimization step. Moreover, $\gamma_t(x)$ can be precomputed once before all iterations.

3.2 Relationship to Smoothing

The original CCM uses additive smoothing in its M-step to capture the fact that distituents outnumber constituents. For each span or context, CCM adds 10 counts: 2 as a constituent and 8 as a distituent.[4] We note that these smoothing parameters are tailored to short sentences: in a binary tree, the number of constituents grows linearly with sentence length, whereas the number of distituents grows quadratically. Therefore, the ratio of constituents to distituents is not constant across sentence lengths. In contrast, by virtue of the log-linear model, LLCCM assigns positive probability to all spans or contexts without explicit smoothing.

[4]These counts are specified in (Klein, 2005); Klein and Manning (2002) added 10 constituent and 50 distituent counts.

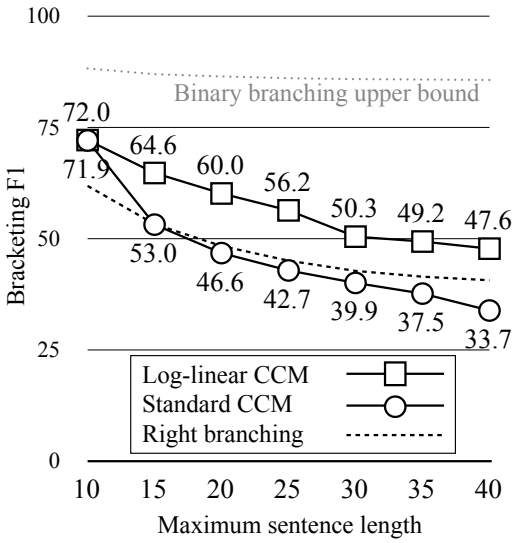


Figure 1: CCM and LLCCM trained and tested on sentences of a fixed length. LLCCM performs well on longer sentences. The binary branching upper bound correponds to UBOUND from (Klein and Manning, 2002).

4 Experiments

We train our models on gold POS sequences from all sections (0-24) of the WSJ (Marcus et al., 1993) with punctuation removed. We report bracketing F1 scores between the binary trees predicted by the models on these sequences and the treebank parses.

We train and evaluate both a CCM implementation (Luque, 2011) and our LLCCM on sentences up to a fixed length n, for $n \in \{10, 15, \ldots, 40\}$. Figure 1 shows that LLCCM substantially outperforms the CCM on longer sentences. After length 15, CCM accuracy falls below the right branching baseline, whereas LLCCM remains significantly better than right-branching through length 40.

5 Conclusion

Our log-linear variant of the CCM extends robustly to long sentences, enabling constituent grammar induction to be used in settings that typically include long sentences, such as machine translation reordering (Chiang, 2005; DeNero and Uszkoreit, 2011; Dyer et al., 2011).

Acknowledgments

We thank Taylor Berg-Kirkpatrick and Dan Klein for helpful discussions regarding the work on which this paper is based. This work was partially supported by the National Science Foundation through a Graduate Research Fellowship to the first author.

References

Taylor Berg-Kirkpatrick and Dan Klein. 2010. Phylogenetic grammar induction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1288–1297, Uppsala, Sweden, July. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick, Alexandre Bouchard-Côté, John DeNero, and Dan Klein. 2010. Painless unsupervised learning with features. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 582–590, Los Angeles, California, June. Association for Computational Linguistics.

Rens Bod. 2006. Unsupervised parsing with U-DOP. In *Proceedings of the Conference on Computational Natural Language Learning*.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. In *Workshop Notes for Statistically-Based NLP Techniques, AAAI*, pages 1–13.

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*, pages 263–270, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Shay B. Cohen and Noah A. Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 74–82, Boulder, Colorado, June. Association for Computational Linguistics.

Shay B. Cohen, Dipanjan Das, and Noah A. Smith. 2011. Unsupervised structure prediction with non-parallel multilingual guidance. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 50–61, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

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)*, 39(1):1–38.

John DeNero and Jakob Uszkoreit. 2011. Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Chris Dyer, Kevin Gimpel, Jonathan H. Clark, and Noah A. Smith. 2011. The CMU-ARK German-English translation system. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 337–343, Edinburgh, Scotland, July. Association for Computational Linguistics.

William P. Headden III, Mark Johnson, and David McClosky. 2009. Improving unsupervised dependency parsing with richer contexts and smoothing. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 101–109, Boulder, Colorado, June. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2002. A generative constituent-context model for improved grammar induction. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 128–135, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics, Main Volume*, pages 478–485, Barcelona, Spain, July.

Dan Klein. 2005. *The Unsupervised Learning of Natural Language Structure*. Ph.D. thesis.

Karim Lari and Steve J. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory method for large scale optimization. *Mathematical Programming B*, 45(3):503–528.

Franco Luque. 2011. Una implementación del modelo DMV+CCM para parsing no supervisado. In *2do Workshop Argentino en Procesamiento de Lenguaje Natural*.

Mitchell P. Marcus, Beatrice Santorini, and Mary A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Tahira Naseem and Regina Barzilay. 2011. Using semantic cues to learn syntax. In *AAAI*.

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1234–1244, Cambridge, MA, October. Association for Computational Linguistics.

Fernando Pereira and Yves 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.

Elias Ponvert, Jason Baldridge, and Katrin Erk. 2011. Simple unsupervised grammar induction from raw text with cascaded finite state models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086, Portland, Oregon, USA, June. Association for Computational Linguistics.

Roi Reichart and Ari Rappoport. 2010. Improved fully unsupervised parsing with zoomed learning. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 684–693, Cambridge, MA, October. Association for Computational Linguistics.

Yoav Seginer. 2007. Fast unsupervised incremental parsing. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 384–391, Prague, Czech Republic, June. Association for Computational Linguistics.

Private Access to Phrase Tables for Statistical Machine Translation

Nicola Cancedda
Xerox Research Centre Europe
6, chemin de Maupertuis
38240, Meylan, France
Nicola.Cancedda@xrce.xerox.com

Abstract

Some Statistical Machine Translation systems never see the light because the owner of the appropriate training data cannot release them, and the potential user of the system cannot disclose what should be translated. We propose a simple and practical encryption-based method addressing this barrier.

1 Introduction

It is generally taken for granted that whoever is deploying a Statistical Machine Translation (SMT) system has unrestricted rights to access and use the parallel data required for its training. This is not always the case. The ideal resources for training SMT models are Translation Memories (TM), especially when they are large, well maintained, coherent in genre and topic and aligned with the application of interest. Such TMs are cherished as valuable assets by their owners, who rarely accept to give away wholesale rights to their use. At the same time, the prospective user of the SMT system that could be derived from such TM might be subject to confidentiality constraints on the text stream needing translation, so that sending out text to translate to an SMT system deployed by the owner of the PT is not an option.

We propose an encryption-based method that addresses such conflicting constraints. In this method, the owner of the TM generates a Phrase Table (PT) from it, and makes it accessible to the user following a special procedure. An SMT decoder is deployed by the user, with all the required resources to operate except the PT[1].

As a result of following the proposed procedure:

- The user acquires all and only the phrase table entries required to perform the decoding of a specific file, thus avoiding complete transfer of the TM to the user;
- The owner of the PT does not learn anything about what is being translated, thus satisfying the user's confidentiality constraints;
- The owner of the PT can track the number of phrase-table entries that was downloaded by the user.

The method assumes that, besides the PT *Owner* and the PT *User*, there is a *Trusted Third Party*. This means that both the User and the PT owner trust such third party not to collude with the other one for violating their secrets (i.e. the content of the PT, or a string requiring translation), even if they do not trust her enough to directly disclose such secrets to her.

While the exposition will focus on phrase tables, there is nothing in the method precluding its use with other resources, provided that they can be represented as look-up tables, a very mild constraint. Provided speed-related aspects can be dealt with, this makes the method directly applicable to language models, or distortion tables for models with lexicalized distortion (Al-Onaizan and Papineni, 2006). The method is also directly applicable to Translation Memories, which can be seen as "degenerate"

[1] If the decoder can operate with multiple PTs, then there could be other (possibly out-of-domain) PTs installed locally.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 23–27,
Jeju, Republic of Korea, 8-14 July 2012.

phrase tables where each record contains only a translation in the target language, and no associated statistics.

The rest of this paper is organized as follows: Section 2 explains the proposed method; in Section 3 we make more precise some implementation choices. We briefly touch on related work on Section 4, provide an experimental validation in Sec. 5, and offer some concluding remarks in Sec. 6.

2 Private access to phrase tables

Let Alice[2] be the owner of a PT, Bob the owner of the SMT decoder who would like to use the table, and Tina a trusted third-party. In broad terms, the proposed method works like this: in an initialization phase, Alice first encrypts PT entries one by one, sends the encrypted PT to Bob, and the encryption/decryption keys to Tina. Alice also sends a method to map source language phrases to PT indices to Bob.

When translating, Bob uses the mapping method sent by Alice to check if a given source phrase is present and has a translation in the PT and, if this is the case, retrieves the index of the corresponding entry in the PT. If the check is positive, then Bob sends a request to Tina for the corresponding decryption key. Tina delivers the decryption key to Bob and communicates that a download has taken place to Alice, who can then increase a download counter.

Let $\{(s_1, v_1), \ldots, (s_n, v_n)\}$ be a PT, where s_i is a source phrase and v_i is the corresponding record. In an actual PT there are multiple lines for a same source phrase, but it is always possible to reconstruct a single record by concatenating all such lines.

2.1 Initialization

The initialization phase is illustrated in Fig. 1. For each PT entry (s_i, v_i), Alice:

1. Encrypts v_i with key k_i We denote the encrypted record as $v_i \oplus k_i$
2. Computes a *digest* d_i of the source entry s_i
3. Sends the phrase digests $\{d_i\}_{i=1,\ldots,n}$ to Bob

[2]We adopt a widespread convention in cryptography and assign person names to the parties involved in the exchange.

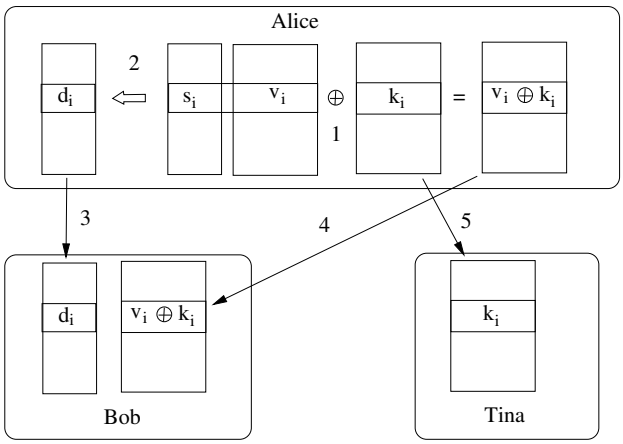


Figure 1: The initialization phase of the method (Sec. 2.1). Bob receives an encrypted version of the PT entries and the corresponding source phrase digests. Tina receives the decryption keys.

4. Sends the encrypted record (or *ciphertext*) $\{v_i \oplus k_i\}_{i=1,\ldots,n}$ to Bob
5. Sends the keys $\{k_i\}_{i=1,\ldots,n}$ to Tina

A *digest*, or *one-way hash function* (Schneider, 1996), is a particular type of hash function. It takes as input a string of arbitrary length, and deterministically produces a bit string of fixed length. It is such that it is virtually impossible to reconstruct a message given its digest, and that the probability of collisions, i.e. of two strings being given the same digest, is negligible.

At the end of the initialization, neither Bob nor Tina can access the content of the PT, unless they collude.

2.2 Retrieval

During translation, Bob has a source phrase s and would like to retrieve from the PT the corresponding entry, if it is present. To do so (Fig. 2):

1. Bob computes the digest d of s using the same cryptographic hash function used by Alice in the initialization phase;
2. Bob checks whether $d \in \{d_i\}_{i=1,\ldots,n}$. If the check is negative then s does not have an entry in the PT, and the process stops. If the check is positive then s has an entry in the PT: let i_s be the corresponding index;

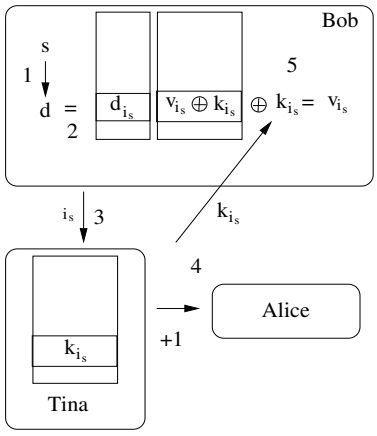


Figure 2: The retrieval phase (Sec. 2.2).

3. Bob requests to Tina key k_{i_s};

4. Tina sends Bob k_{i_s} and notifies Alice, who can increment a counter of PT entries downloaded by Bob;

5. Bob decrypts $v_{i_s} \oplus k_{i_s}$ using key k_{i_s}, and recovers v_{i_s}.

At the end of the process, Bob retrieved from the PT owned by Alice an entry if and only if it matched phrase s (this is guaranteed by the virtual absence of collisions ensured by the cryptographic hash functions used for computing phrase digests). Alice was notified by Tina that Bob downloaded one entry, as desired, while neither Tina nor Alice could learn s, unless they colluded.

3 Implementation

For clarity of exposition, in Section 2.2 we presented a method for looking up PT entries involving one interaction for each phrase look-up. In our implementation, we batch all requests for all source phrases up to a predefined length for all sentences in a given file. This mirrors the standard practice of filtering the phrase table for a given source file to translate before starting the actual decoding.

Out of the large choice of cryptographic hash functions in the literature (Schneider, 1996), we chose 128 bits *md5* for its widespread availability in multiple programming languages and environments.

For encrypting entries, we used bit-wise XOR with a string of random bits (the key) of the same length as the encrypted item. This symmetric encryption is known as *one-time pad*, and it is unbreakable, provided key bits are really random.

Both keys and ciphertext are indexed and sorted by increasing md5 digest of the corresponding source phrase. For retrieving all entries matching a given text file, Bob generates md5 digests for all source phrases up to a maximum length, sorts them, and performs a *join* with the encrypted entry file. Matching digests are then sent to Tina for her to join with the keys. It is important that Bob uses the same tokenizer/word segmentation scheme used by Alice in preprocessing training data before extracting the PT.

Note that it is never necessary to have any massive data structure in main memory, and all process steps except the initial sorting by md5 digest are linear in the number of PT entries or in the number of tokens to look up. The process results however in increased storage and bandwidth requirements, since ciphertext and key have each roughly the same size as the original PT.

4 Related work

We are not aware of any previous work directly addressing the problem we solve, i.e. private access to a phrase table or other resources for the purpose of performing statistical machine translation. Private access to electronic information in general, however, is an active research area. While effective, the scheme proposed here is rather basic, compared to what can be found in specialized literature, e.g. (Chor et al., 1998; Bellovin and Cheswick, 2004). An interesting and relatively recent survey of the field of secure multiparty computation and privacy-preserving data mining is (Lindell and Pinkas, 2009).

5 Experiments

We validated our simple implementation using a phrase table of 38,488,777 lines created with the Moses toolkit[3](Koehn et al., 2007) phrase-based SMT system, corresponding to 15,764,069 entries for distinct source phrases[4].

[3]http://www.statmt.org/moses/

[4]The *birthday bound* for a 128 bit hash like md5 for a collision probability of 10^{-18} is around $2.6 * 10^{10}$. This means

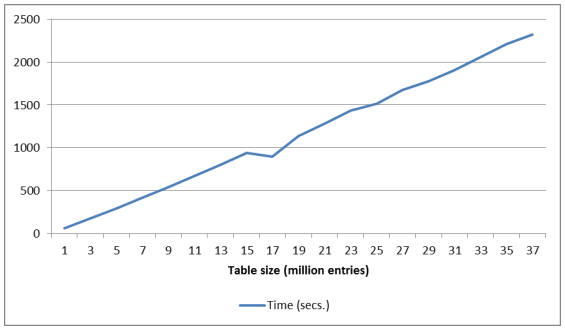


Figure 3: Time required to complete the initialization as a function of the number of lines in the original PT.

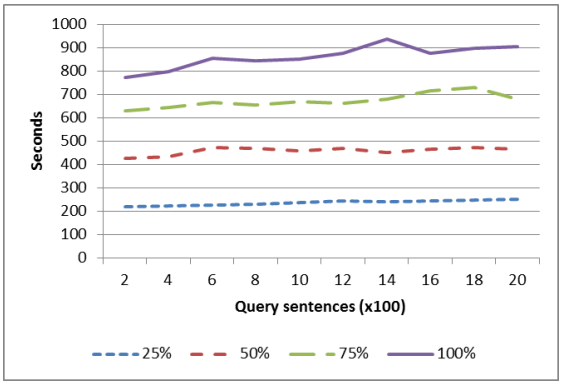


Figure 4: Time required for retrieval as a function of the number of sentences in the query, for different subsets of the original phrase table.

This PT was obtained processing the training data of the English-Spanish Europarl corpus used in the WMT 2008 shared task[5]. We used a 2,000 sentence test set of the same shared evaluation for experimenting with the querying phase.

We conducted all experiments on a single core of an ordinary Linux server[6] with 32Gb of RAM. Both initialization and retrieval can be easily parallelized.

Figure 3 shows the time required to complete the initialization phase as a function of the size of the original PT (in million of lines). The progression is largely linear, and the overall initialization time of roughly 45 minutes for the complete PT indicates that the method can be used in practice. Note that the Europarl corpus originating the phrase-table is much larger than most TMs available at even large language service providers.

Figure 4 displays the time required to complete retrieval for subsets of increasing size of the 2,000 sentence test set, and for phrase tables uniformly sampled at 25%, 50%, 75% and 100%. 217,019 distinct digests are generated for all possible phrase of length up to 6 from the full test set, resulting in the retrieval of 47,072 entries (596,560 lines) from the full phrase table. Our implementation of the retrieval uses the Unix *join* command on the ciphertext and the key tables, and performs a full scan through those files. Complexity hence depends more on the size of the PT than on the length of the query. An ad-hoc indexing of the encrypted entries and of the keys in e.g. a standard database would make the dependency logarithmic in the number of entries, and linear in the number of source tokens. Digests' prefixes are perfectly suited for bucketing ciphertext and keys. This would be useful if query batches are small.

6 Conclusions

Some SMT systems never get deployed because of legitimate and incompatible concerns of the prospective users and of the training data owners. We propose a method that guarantees to the owner of a TM that only some fraction of an artifact derived from the original resource, a phrase-table, is transferred, and only in a very controlled way allowing to track downloads. This same method also guarantees the privacy of the user, who is not required to disclose the content of what needs translation.

Empirical validation on demanding conditions shows that the proposed method is practical on ordinary computing infrastructure.

This same method can be easily extended to other resources used by SMT systems, and indeed even beyond SMT itself, whenever similar constraints on data access exist.

that if the hash distributed keys perfectly uniformly, then about 26 billion entries would be required for the collision probability to exceed 10^{-18}. While no hash function, including md5, distributes keys *perfectly* evenly (Bellare and Kohno, 2004), the number of entries likely to be handled in our application is orders of magnitude smaller than the bound.

[5]http://www.statmt.org/wmt08/shared-task.html

[6]Intel Xeon 3.1 GHz.

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 the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 529–536, Stroudsburg, PA, USA. Association for Computational Linguistics.

Mihir Bellare and Tadayoshi Kohno. 2004. Hash function balance and its impact on birthday attacks. In *Advances in Cryptology, EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 401–418.

Steven M. Bellovin and William R. Cheswick. 2004. Privacy-enhanced searches using encrypted bloom filters. Technical Report CUCS-034-07, Columbia University.

Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. 1998. Private information retrieval. *Journal of the ACM*, 45(6):965–982.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yehuda Lindell and Benny Pinkas. 2009. Secure multiparty computation and privacy-preserving data mining. *The Journal of Privacy and Confidentiality*, 1(1):59–98.

Bruce Schneider. 1996. *Applied Cryptography*. John Wiley and sons.

Fast and Scalable Decoding with Language Model Look-Ahead for Phrase-based Statistical Machine Translation

Joern Wuebker, Hermann Ney
Human Language Technology and Pattern Recognition Group
Computer Science Department
RWTH Aachen University, Germany
surname@cs.rwth-aachen.de

Richard Zens*
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
zens@google.com

Abstract

In this work we present two extensions to the well-known dynamic programming beam search in phrase-based statistical machine translation (SMT), aiming at increased efficiency of decoding by minimizing the number of language model computations and hypothesis expansions. Our results show that language model based pre-sorting yields a small improvement in translation quality and a speedup by a factor of 2. Two look-ahead methods are shown to further increase translation speed by a factor of 2 without changing the search space and a factor of 4 with the side-effect of some additional search errors. We compare our approach with Moses and observe the same performance, but a substantially better trade-off between translation quality and speed. At a speed of roughly 70 words per second, Moses reaches 17.2% BLEU, whereas our approach yields 20.0% with identical models.

1 Introduction

Research efforts to increase search efficiency for phrase-based MT (Koehn et al., 2003) have explored several directions, ranging from generalizing the stack decoding algorithm (Ortiz et al., 2006) to additional early pruning techniques (Delaney et al., 2006), (Moore and Quirk, 2007) and more efficient language model (LM) querying (Heafield, 2011).

This work extends the approach by (Zens and Ney, 2008) with two techniques to increase translation speed and scalability. We show that taking a heuristic LM score estimate for pre-sorting the phrase translation candidates has a positive effect on both translation quality and speed. Further, we introduce two novel LM look-ahead methods. The idea of LM look-ahead is to incorporate the LM probabilities into the pruning process of the beam search as early as possible. In speech recognition it has been used for many years (Steinbiss et al., 1994; Ortmanns et al., 1998). *First-word LM look-ahead* exploits the search structure to use the LM costs of the first word of a new phrase as a lower bound for the full LM costs of the phrase. *Phrase-only LM look-ahead* makes use of a pre-computed estimate of the full LM costs for each phrase. We detail the implementation of these methods and analyze their effect with respect to the number of LM computations and hypothesis expansions as well as on translation speed and quality. We also run comparisons with the Moses decoder (Koehn et al., 2007), which yields the same performance in BLEU, but is outperformed significantly in terms of scalability for faster translation. Our implementation is available under a non-commercial open source licence[†].

2 Search Algorithm Extensions

We apply the decoding algorithm described in (Zens and Ney, 2008). Hypotheses are scored by a weighted log-linear combination of models. A beam search strategy is used to find the best hypothesis. During search we perform pruning controlled by the parameters *coverage histogram size*[‡] N_c and *lexical*

*Richard Zens's contribution was during his time at RWTH.
[†] www-i6.informatik.rwth-aachen.de/jane
[‡] number of hypothesized coverage vectors per cardinality

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 28–32,
Jeju, Republic of Korea, 8-14 July 2012.

histogram size[§] N_l.

2.1 Phrase candidate pre-sorting

In addition to the source sentence f_1^J, the beam search algorithm takes a matrix $E(\cdot,\cdot)$ as input, where for each contiguous phrase $\tilde{f} = f_j \dots f_{j'}$ within the source sentence, $E(j, j')$ contains a list of all candidate translations for $\tilde{f}$. The candidate lists are sorted according to their model score, which was observed to speed up translation by Delaney et al. (2006). In addition to sorting according to the purely phrase-internal scores, which is common practice, we compute an estimate $q_{\mathrm{LME}}(\tilde{e})$ for the LM score of each target phrase $\tilde{e}$. $q_{\mathrm{LME}}(\tilde{e})$ is the weighted LM score we receive by assuming $\tilde{e}$ to be a complete sentence without using sentence start and end markers. We limit the number of translation options per source phrase to the N_o top scoring candidates (observation histogram pruning).

The pre-sorting during phrase matching has two effects on the search algorithm. Firstly, it defines the order in which the hypothesis expansions take place. As higher scoring phrases are considered first, it is less likely that already created partial hypotheses will have to be replaced, thus effectively reducing the expected number of hypothesis expansions. Secondly, due to the observation pruning the sorting affects the considered phrase candidates and consequently the search space. A better pre-selection can be expected to improve translation quality.

2.2 Language Model Look-Ahead

LM score computations are among the most expensive in decoding. Delaney et al. (2006) report significant improvements in runtime by removing unnecessary LM lookups via early pruning. Here we describe an LM look-ahead technique, which is aimed at further reducing the number of LM computations.

The innermost loop of the search algorithm iterates over all translation options for a single source phrase to consider them for expanding the current hypothesis. We introduce an LM look-ahead score $q_{\mathrm{LMLA}}(\tilde{e}|\tilde{e}')$, which is computed for each of the translation options. This score is added to the overall hypothesis score, and if the pruning threshold is exceeded, we discard the expansion without computing the full LM score.

First-word LM look-ahead pruning defines the LM look-ahead score $q_{\mathrm{LMLA}}(\tilde{e}|\tilde{e}') = q_{\mathrm{LM}}(\tilde{e}_1|\tilde{e}')$ to be the LM score of the first word of target phrase $\tilde{e}$ given history $\tilde{e}'$. As $q_{\mathrm{LM}}(\tilde{e}_1|\tilde{e}')$ is an upper bound for the full LM score, the technique does not introduce additional seach errors. The score can be reused, if the LM score of the full phrase $\tilde{e}$ needs to be computed afterwards.

We can exploit the structure of the search to speed up the LM lookups for the first word. The LM probabilities are stored in a *trie*, where each node corresponds to a specific LM history. Usually, each LM lookup consists of first traversing the trie to find the node corresponding to the current LM history and then retrieving the probability for the next word. If the n-gram is not present, we have to repeat this procedure with the next lower-order history, until a probability is found. However, the LM history for the first words of all phrases within the innermost loop of the search algorithm is identical. Just before the loop we can therefore traverse the trie once for the current history and each of its lower order n-grams and store the pointers to the resulting nodes. To retrieve the LM look-ahead scores, we can then directly access the nodes without the need to traverse the trie again. This implementational detail was confirmed to increase translation speed by roughly 20% in a short experiment.

Phrase-only LM look-ahead pruning defines the look-ahead score $q_{\mathrm{LMLA}}(\tilde{e}|\tilde{e}') = q_{\mathrm{LME}}(\tilde{e})$ to be the LM score of phrase $\tilde{e}$, assuming $\tilde{e}$ to be the full sentence. It was already used for sorting the phrases, is therefore pre-computed and does not require additional LM lookups. As it is not a lower bound for the real LM score, this pruning technique can introduce additional search errors. Our results show that it radically reduces the number of LM lookups.

3 Experimental Evaluation

3.1 Setup

The experiments are carried out on the German→English task provided for WMT 2011[*].

[§] number of lexical hypotheses per coverage vector

[*] http://www.statmt.org/wmt11

system	BLEU[%]	#HYP	#LM	w/s
$N_o = \infty$				
baseline	20.1	3.0K	322K	2.2
+pre-sort	20.1	2.5K	183K	3.6
$N_o = 100$				
baseline	19.9	2.3K	119K	7.1
+pre-sort	20.1	1.9K	52K	15.8
+first-word	20.1	1.9K	40K	31.4
+phrase-only	19.8	1.6K	6K	69.2

Table 1: Comparison of the number of hypothesis expansions per source word (#HYP) and LM computations per source word (#LM) with respect to LM pre-sorting, first-word LM look-ahead and phrase-only LM look-ahead on `newstest2009`. Speed is given in words per second. Results are given with ($N_o = 100$) and without ($N_o = \infty$) observation pruning.

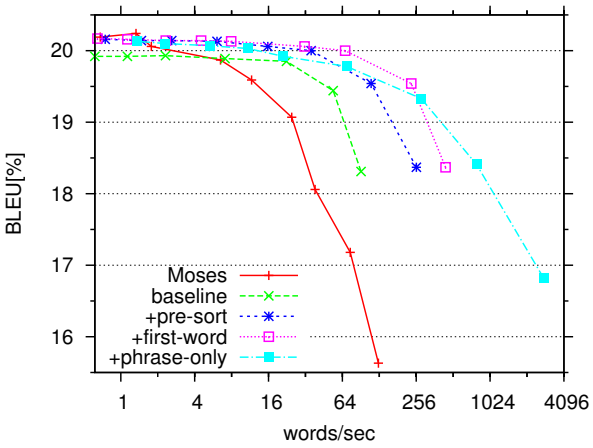


Figure 1: Translation performance in BLEU [%] on the `newstest2009` set vs. speed on a logarithmic scale. We compare Moses with our approach without LM look-ahead and LM score pre-sorting (baseline), with added LM pre-sorting and with either first-word or phrase-only LM look-ahead on top of +pre-sort. Observation histogram size is fixed to $N_o = 100$ for both decoders.

The English language model is a 4-gram LM created with the SRILM toolkit (Stolcke, 2002) on all bilingual and parts of the provided monolingual data. `newstest2008` is used for parameter optimization, `newstest2009` as a blind test set. To confirm our results, we run the final set of experiments also on the English→French task of IWSLT 2011[†]. We evaluate with BLEU (Papineni et al., 2002) and TER (Snover et al., 2006).

We use identical phrase tables and scaling factors for Moses and our decoder. The phrase table is pruned to a maximum of 400 target candidates per source phrase before decoding. The phrase table and LM are loaded into memory before translating and loading time is eliminated for speed measurements.

3.2 Methodological analysis

To observe the effect of the proposed search algorithm extensions, we ran experiments with fixed pruning parameters, keeping track of the number of hypothesis expansions and LM computations. The LM score pre-sorting affects both the set of phrase candidates due to observation histogram pruning and the order in which they are considered. To separate these effects, experiments were run both with histogram pruning ($N_o = 100$) and without. From Table 1 we can see that in terms of efficiency both cases show similar improvements over the baseline, which performs pre-sorting with respect to the translation model scores only. The number of hypothesis expansions is reduced by ∼20% and the number of LM lookups by ∼50%. When observation pruning is applied, we additionally observe a small increase by 0.2% in BLEU.

Application of first-word LM look-ahead further reduces the number of LM lookups by 23%, resulting in doubled translation speed, part of which derives from fewer trie node searches. The heuristic phrase-only LM look-ahead method introduces additional search errors, resulting in a BLEU drop by 0.3%, but yields another 85% reduction in LM computations and increases throughput by a factor of 2.2.

3.3 Performance evaluation

In this section we evaluate the proposed extensions to the original beam search algorithm in terms of scalability and their usefulness for different application constraints. We compare Moses and four different setups of our decoder: LM score pre-sorting switched on or off without LM look-ahead and both LM look-ahead methods with LM score pre-sorting. We translated the test set with the beam sizes set to $N_c = N_l = \{1,2,4,8,16,24,32,48,64\}$. For Moses we used the beam sizes $2^i, i \in \{1,\ldots,9\}$. Transla-

[†] `http://iwslt2011.org`

setup	system	WMT 2011 German→English				IWSLT 2011 English→French			
		beam size (N_c,N_l)	speed w/s	BLEU [%]	TER [%]	beam size (N_c,N_l)	speed w/s	BLEU [%]	TER [%]
best	Moses	256	0.7	20.2	63.2	16	10	29.5	52.8
	this work: first-word	(48,48)	1.1	20.2	63.3	(8,8)	23	29.5	52.9
	phrase-only	(64,64)	1.4	20.1	63.2	(16,16)	18	29.5	52.8
BLEU:	Moses	16	12	19.6	63.7	4	40	29.1	53.2
$\geq$ -1%	this work: first-word	(4,4)	67	20.0	63.2	(2,2)	165	29.1	53.1
	phrase-only	(8,8)	69	19.8	63.0	(4,4)	258	29.3	52.9
BLEU:	Moses	8	25	19.1	64.2	2	66	28.1	54.3
$\geq$ -2%	this work: first-word	(2,2)	233	19.5	63.4	(1,1)	525	28.4	53.9
	phrase-only	(4,4)	280	19.3	63.0	(2,2)	771	28.5	53.2
fastest	Moses	1	126	15.6	68.3	1	116	26.7	55.9
	this work: first-word	(1,1)	444	18.4	64.6	(1,1)	525	28.4	53.9
	phrase-only	(1,1)	2.8K	16.8	64.4	(1,1)	2.2K	26.4	54.7

Table 2: Comparison of Moses with this work. Either first-word or phrase-only LM look-ahead is applied. We consider both the best and the fastest possible translation, as well as the fastest settings resulting in no more than 1% and 2% BLEU loss on the development set. Results are given on the test set (`newstest2009`).

tion performance in BLEU is plotted against speed in Figure 1. Without the proposed extensions, Moses slightly outperforms our decoder in terms of BLEU. However, the latter already scales better for higher speed. With LM score pre-sorting, the best BLEU value is similar to Moses while further accelerating translation, yielding identical performance at 16 words/sec as Moses at 1.8 words/sec. Application of first-word LM look-ahead shifts the graph to the right, now reaching the same performance at 31 words/sec. At a fixed translation speed of roughly 70 words/sec, our approach yields 20.0% BLEU, whereas Moses reaches 17.2%. For phrase-only LM look-ahead the graph is somewhat flatter. It yields nearly the same top performance with an even better trade-off between translation quality and speed.

The final set of experiments is performed on both the WMT and the IWSLT task. We directly compare our decoder with the two LM look-ahead methods with Moses in four scenarios: the best possible translation, the fastest possible translation without performance constraint and the fastest possible translation with no more than 1% and 2% loss in BLEU on the dev set compared to the best value. Table 2 shows that on the WMT data, the top performance is similar for both decoders. However, if we allow for a small degradation in translation performance, our approaches clearly outperform Moses in terms of translation speed. With phrase-only LM look-ahead, our decoder is faster by a factor of 6 for no more than 1% BLEU loss, a factor of 11 for 2% BLEU loss and a factor of 22 in the fastest setting. The results on the IWSLT data are very similar. Here, the speed difference reaches a factor of 19 in the fastest setting.

4 Conclusions

This work introduces two extensions to the well-known beam search algorithm for phrase-based machine translation. Both pre-sorting the phrase translation candidates with an LM score estimate and LM look-ahead during search are shown to have a positive effect on translation speed. We compare our decoder to Moses, reaching a similar highest BLEU score, but clearly outperforming it in terms of scalability with respect to the trade-off ratio between translation quality and speed. In our experiments, the fastest settings of our decoder and Moses differ in translation speed by a factor of 22 on the WMT data and a factor of 19 on the IWSLT data. Our software is part of the open source toolkit *Jane*.

Acknowledgments

This work was partially realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

References

[Delaney et al.2006] Brian Delaney, Wade Shen, and Timothy Anderson. 2006. An efficient graph search decoder for phrase-based statistical machine translation. In *International Workshop on Spoken Language Translation*, Kyoto, Japan, November.

[Heafield2011] Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the 6th Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, UK, July.

[Koehn et al.2003] P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 Meeting of the North American chapter of the Association for Computational Linguistics (NAACL-03)*, pages 127–133, Edmonton, Alberta.

[Koehn et al.2007] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantine, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session*, pages 177–180, Prague, Czech Republic, June.

[Moore and Quirk2007] Robert C. Moore and Chris Quirk. 2007. Faster beam-search decoding for phrasal statistical machine translation. In *Proceedings of MT Summit XI*.

[Ortiz et al.2006] Daniel Ortiz, Ismael Garcia-Varea, and Francisco Casacuberta. 2006. Generalized stack decoding algorithms for statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 64–71, New York City, June.

[Ortmanns et al.1998] S. Ortmanns, H. Ney, and A. Eiden. 1998. Language-model look-ahead for large vocabulary speech recognition. In *International Conference on Spoken Language Processing*, pages 2095–2098, Sydney, Australia, October.

[Papineni et al.2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July.

[Snover et al.2006] 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.

[Steinbiss et al.1994] V. Steinbiss, B. Tran, and Hermann Ney. 1994. Improvements in Beam Search. In *Proc. of the Int. Conf. on Spoken Language Processing (ICSLP'94)*, pages 2143–2146, September.

[Stolcke2002] Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. In *Proceedings of the Seventh International Conference on Spoken Language Processing*, pages 901–904. ISCA, September.

[Zens and Ney2008] Richard Zens and Hermann Ney. 2008. Improvements in Dynamic Programming Beam Search for Phrase-based Statistical Machine Translation. In *International Workshop on Spoken Language Translation*, pages 195–205, Honolulu, Hawaii, October.

Head-Driven Hierarchical Phrase-based Translation

Junhui Li Zhaopeng Tu[†] Guodong Zhou[‡] Josef van Genabith
Centre for Next Generation Localisation
School of Computing, Dublin City University
[†] Key Lab. of Intelligent Info. Processing
Institute of Computing Technology, Chinese Academy of Sciences
[‡]School of Computer Science and Technology
Soochow University, China
{jli,josef}@computing.dcu.ie
tuzhaopeng@ict.ac.cn gdzhou@suda.edu.cn

Abstract

This paper presents an extension of Chiang's hierarchical phrase-based (HPB) model, called Head-Driven HPB (HD-HPB), which incorporates head information in translation rules to better capture syntax-driven information, as well as improved reordering between any two neighboring non-terminals at any stage of a derivation to explore a larger reordering search space. Experiments on Chinese-English translation on four NIST MT test sets show that the HD-HPB model significantly outperforms Chiang's model with average gains of 1.91 points absolute in BLEU.

1 Introduction

Chiang's hierarchical phrase-based (HPB) translation model utilizes synchronous context free grammar (SCFG) for translation derivation (Chiang, 2005; Chiang, 2007) and has been widely adopted in statistical machine translation (SMT). Typically, such models define two types of translation rules: hierarchical (translation) rules which consist of both terminals and non-terminals, and glue (grammar) rules which combine translated phrases in a monotone fashion. Due to lack of linguistic knowledge, Chiang's HPB model contains only one type of non-terminal symbol X, often making it difficult to select the most appropriate translation rules.[1] What is more, Chiang's HPB model suffers from limited phrase reordering combining translated phrases in a monotonic way with glue rules. In addition, once a glue rule is adopted, it requires all rules above it to be glue rules.

One important research question is therefore how to refine the non-terminal category *X* using linguistically motivated information: Zollmann and Venugopal (2006) (SAMT) e.g. use (partial) syntactic categories derived from CFG trees while Zollmann and Vogel (2011) use word tags, generated by either POS analysis or unsupervised word class induction. Almaghout et al. (2011) employ CCG-based supertags. Mylonakis and Sima'an (2011) use linguistic information of various granularities such as *Phrase-Pair*, *Constituent*, *Concatenation of Constituents*, and *Partial Constituents*, where applicable. Inspired by previous work in parsing (Charniak, 2000; Collins, 2003), our Head-Driven HPB (HD-HPB) model is based on the intuition that linguistic heads provide important information about a constituent or distributionally defined fragment, as in HPB. We identify heads using linguistically motivated dependency parsing, and use their POS to refine *X*. In addition HD-HPB provides flexible reordering rules freely mixing translation and reordering (including swap) at any stage in a derivation.

Different from the soft constraint modeling adopted in (Chan et al., 2007; Marton and Resnik, 2008; Shen et al., 2009; He et al., 2010; Huang et al., 2010; Gao et al., 2011), our approach encodes syntactic information in translation rules. However, the two approaches are not mutually exclusive, as we could also include a set of syntax-driven features into our translation model. Our approach maintains the advantages of Chiang's HPB model while at the same time incorporating head information and flex-

[1] Another non-terminal symbol S is used in glue rules.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 33–37,
Jeju, Republic of Korea, 8-14 July 2012.

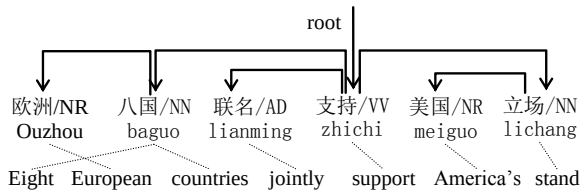


Figure 1: An example word alignment for a Chinese-English sentence pair with the dependency parse tree for the Chinese sentence. Here, each Chinese word is attached with its POS tag and Pinyin.

ible reordering in a derivation in a natural way. Experiments on Chinese-English translation using four NIST MT test sets show that our HD-HPB model significantly outperforms Chiang's HPB as well as a SAMT-style refined version of HPB.

2 Head-Driven HPB Translation Model

Like Chiang (2005) and Chiang (2007), our HD-HPB translation model adopts a synchronous context free grammar, a rewriting system which generates source and target side string pairs simultaneously using a context-free grammar. Instead of collapsing all non-terminals in the source language into a single symbol X as in Chiang (2007), given a word sequence f_j^i from position i to position j, we first find **heads** and then concatenate the POS tags of these heads as f_j^i's non-terminal symbol. Specifically, we adopt unlabeled dependency structure to derive heads, which are defined as:

Definition 1. *For word sequence f_j^i, word $f_k\,(i \leq k \leq j)$ is regarded as a **head** if it is dominated by a word outside of this sequence.*

Note that this definition (i) allows for a word sequence to have one or more heads (largely due to the fact that a word sequence is not necessarily linguistically constrained) and (ii) ensures that heads are always the highest heads in the sequence from a dependency structure perspective. For example, the word sequence *ouzhou baguo lianming* in Figure 1 has two heads (i.e., *baguo* and *lianming*, *ouzhou* is not a head of this sequence since its headword *baguo* falls within this sequence) and the non-terminal corresponding to the sequence is thus labeled as *NN-AD*. It is worth noting that in this paper we only refine non-terminal X on the source side to head-informed ones, while still using X on the target side.

According to the occurrence of terminals in translation rules, we group rules in the HD-HPB model into two categories: head-driven hierarchical rules (HD-HRs) and non-terminal reordering rules (NRRs), where the former have at least one terminal on both source and target sides and the later have no terminals. For rule extraction, we first identify *initial phrase pairs* on word-aligned sentence pairs by using the same criterion as most phrase-based translation models (Och and Ney, 2004) and Chiang's HPB model (Chiang, 2005; Chiang, 2007). We extract HD-HRs and NRRs based on initial phrase pairs, respectively.

2.1 HD-HRs: Head-Driven Hierarchical Rules

As mentioned, a HD-HR has at least one terminal on both source and target sides. This is the same as the hierarchical rules defined in Chiang's HPB model (Chiang, 2007), except that we use head POS-informed non-terminal symbols in the source language. We look for initial phrase pairs that contain other phrases and then replace sub-phrases with POS tags corresponding to their heads. Given the word alignment in Figure 1, Table 1 demonstrates the difference between hierarchical rules in Chiang (2007) and HD-HRs defined here.

Similar to Chiang's HPB model, our HD-HPB model will result in a large number of rules causing problems in decoding. To alleviate these problems, we filter our HD-HRs according to the same constraints as described in Chiang (2007). Moreover, we discard rules that have non-terminals with more than four heads.

2.2 NRRs: Non-terminal Reordering Rules

NRRs are translation rules without terminals. Given an initial phrase pair on the source side, there are four possible positional relationships for their target side translations (we use *Y* as a variable for non-terminals on the source side while all non-terminals on the target side are labeled as *X*):

- Monotone $\langle Y \rightarrow Y_1Y_2, X \rightarrow X_1X_2 \rangle$;
- Discontinuous monotone $\langle Y \rightarrow Y_1Y_2, X \rightarrow X_1 \ldots X_2 \rangle$;
- Swap $\langle Y \rightarrow Y_1Y_2, X \rightarrow X_2X_1 \rangle$;
- Discontinuous swap $\langle Y \rightarrow Y_1Y_2, X \rightarrow X_2 \ldots X_1 \rangle$.

phrase pairs	hierarchical rule	head-driven hierarchical rule
lichang, stand	X→lichang, stand	NN→lichang, X→stand
meiguo lichang$_1$, America's stand$_1$	X→meiguo X$_1$, America's X$_1$	NN→meiguo NN$_1$, X→America's X$_1$
zhichi meiguo, support America's	X→zhichi meiguo, support America's	VV-NR→zhichi meiguo, X→support America's
zhichi meiguo$_1$ lichang, support America's$_1$ stand	X→X$_1$ lichang, X$_1$ stand	VV→VV-NR$_1$ lichang, X→X$_1$ stand

Table 1: Comparison of hierarchical rules in Chiang (2007) and HD-HRs. Indexed underlines indicate sub-phrases and corresponding non-terminal symbols. The non-terminals in HD-HRs (e.g., NN, VV, VV-NR) capture the head(s) POS tags of the corresponding word sequence in the source language.

Merging two neighboring non-terminals into a single non-terminal, NRRs enable the translation model to explore a wider search space. During training, we extract four types of NRRs and calculate probabilities for each type. To speed up decoding, we currently (i) only use monotone and swap NRRs and (ii) limit the number of non-terminals in a NRR to 2.

2.3 Features and Decoding

Given e for the translation output in the target language, s and t for strings of terminals and non-terminals on the source and target side, respectively, we use a feature set analogous to the default feature set of Chiang (2007), including:

- $P_{hd\text{-}hr}(t|s)$ and $P_{hd\text{-}hr}(s|t)$, translation probabilities for HD-HRs;
- $P_{lex}(t|s)$ and $P_{lex}(s|t)$, lexical translation probabilities for HD-HRs;
- $Pty_{hd\text{-}hr} = exp(-1)$, rule penalty for HD-HRs;
- $P_{nrr}(t|s)$, translation probability for NRRs;
- $Pty_{nrr} = exp(-1)$, rule penalty for NRRs;
- $P_{lm}(e)$, language model;
- $Pty_{word}(e) = exp(-|e|)$, word penalty.

Our decoder is based on CKY-style chart parsing with beam search and searches for the best derivation bottom-up. For a source span $[i, j]$, it applies both types of HD-HRs and NRRs. However, HD-HRs are only applied to generate derivations spanning no more than K words – the initial phrase length limit used in training to extract HD-HRs – while NRRs are applied to derivations spanning any length. Unlike in Chiang's HPB model, it is possible for a non-terminal generated by a NRR to be included afterwards by a HD-HR or another NRR.

3 Experiments

We evaluate the performance of our HD-HPB model and compare it with our implementation of Chiang's HPB model (Chiang, 2007), a source-side SAMT-style refined version of HPB (SAMT-HPB), and the Moses implementation of HPB. For fair comparison, we adopt the same parameter settings for our HD-HPB and HPB systems, including initial phrase length (as 10) in training, the maximum number of non-terminals (as 2) in translation rules, maximum number of non-terminals plus terminals (as 5) on the source, beam threshold β (as 10^{-5}) (to discard derivations with a score worse than β times the best score in the same chart cell), beam size b (as 200) (i.e. each chart cell contains at most b derivations). For Moses HPB, we use "grow-diag-final-and" to obtain symmetric word alignments, 10 for the maximum phrase length, and the recommended default values for all other parameters.

We train our model on a dataset with ˜1.5M sentence pairs from the LDC dataset.[2] We use the 2002 NIST MT evaluation test data (878 sentence pairs) as the development data, and the 2003, 2004, 2005, 2006-news NIST MT evaluation test data (919, 1788, 1082, and 616 sentence pairs, respectively) as the test data. To find heads, we parse the source sentences with the Berkeley Parser[3] (Petrov and Klein, 2007) trained on Chinese TreeBank 6.0 and use the Penn2Malt toolkit[4] to obtain (unlabeled) dependency structures.

We obtain the word alignments by running

[2]This dataset includes LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06

[3]http://code.google.com/p/berkeleyparser/

[4]http://w3.msi.vxu.se/˜nivre/research/Penn2Malt.html/

GIZA++ (Och and Ney, 2000) on the corpus in both directions and applying "grow-diag-final-and" refinement (Koehn et al., 2003). We use the SRI language modeling toolkit to train a 5-gram language model on the Xinhua portion of the Gigaword corpus and standard MERT (Och, 2003) to tune the feature weights on the development data.

For evaluation, the NIST BLEU script (version 12) with the default settings is used to calculate the BLEU scores. To test whether a performance difference is statistically significant, we conduct significance tests following the paired bootstrap approach (Koehn, 2004). In this paper, ' ** ' and ' * ' denote p-values less than 0.01 and in-between [0.01, 0.05), respectively.

Table 2 lists the rule table sizes. The full rule table size (including HD-HRs and NRRs) of our HD-HPB model is ˜1.5 times that of Chiang's, largely due to refining the non-terminal symbol X in Chiang's model into head-informed ones in our model. It is also unsurprising, that the test set-filtered rule table size of our model is only ˜0.7 times that of Chiang's: this is due to the fact that some of the refined translation rule patterns required by the test set are unattested in the training data. Furthermore, the rule table size of NRRs is much smaller than that of HD-HRs since a NRR contains only two non-terminals.

Table 3 lists the translation performance with BLEU scores. Note that our re-implementation of Chiang's original HPB model performs on a par with Moses HPB. Table 3 shows that our HD-HPB model significantly outperforms Chiang's HPB model with an average improvement of 1.91 in BLEU (and similar improvements over Moses HPB).

Table 3 shows that the head-driven scheme outperforms a SAMT-style approach (for each test set $p < 0.01$), indicating that head information is more effective than (partial) CFG categories. Taking *lianming zhichi* in Figure 1 as an example, HD-HPB labels the span *VV*, as *lianming* is dominated by *zhichi*, effecively ignoring *lianming* in the translation rule, while the SAMT label is *ADVP:AD+VV*[5] which is more susceptible to data sparsity. In addition, SAMT resorts to *X* if a text span fails to satisify pre-defined categories. Examining initial phrases extracted from the SAMT training data shows that 28% of them are labeled as *X*.

[5]the constituency structure for *lianming zhichi* is *(VP (ADVP (AD lianming)) (VP (VV zhichi) ...))*.

System	Total	MT 03	MT 04	MT 05	MT 06	Avg.
HPB	39.6	2.8	4.7	3.3	3.0	3.4
HD-HPB	59.5/0.6	1.9/0.1	3.4/0.2	2.3/0.2	2.0/0.1	2.4/0.2

Table 2: Rule table sizes (in million) of different models. Note: 1) For HD-HPB, the rule sizes separated by / indicate HD-HRs and NRRs, respectively; 2) Except for "Total", the figures correspond to rules filtered on the corresponding test set.

System	MT 03	MT 04	MT 05	MT 06	Avg.
Moses HPB	32.94*	35.16	32.18	29.88*	32.54
HPB	33.59	35.39	32.20	30.60	32.95
HD-HPB	**35.50****	**37.61****	**34.56****	**31.78****	**34.86**
SAMT-HPB	34.07	36.52**	32.90*	30.66	33.54
HD-HR+Glue	34.58**	36.55**	33.84**	31.06	34.01

Table 3: BLEU (%) scores of different models. Note: 1) SAMT-HPB indicates our HD-HPB model with non-terminal scheme of Zollmann and Venugopal (2006); 2) HD-HR+Glue indicates our HD-HPB model replacing NRRs with glue rules; 3) Significance tests for Moses HPB, HD-HPB, SAMT-HPB, and HD-HR+Glue are done against HPB.

In order to separate out the individual contributions of the novel HD-HRs and NRRs, we carry out an additional experiment (HD-HR+Glue) using HD-HRs with monotonic glue rules only (adjusted to refined rule labels, but effectively switching off the extra reordering power of full NRRs). Table 3 shows that on average more than half of the improvement over HPB (Chiang and Moses) comes from the refined HD-HRs, the rest from NRRs.

Examining translation rules extracted from the training data shows that there are 72,366 types of non-terminals with respect to 33 types of POS tags. On average each sentence employs 16.6/5.2 HD-HRs/NRRs in our HD-HPB model, compared to 15.9/3.6 hierarchical rules/glue rules in Chiang's model, providing further indication of the importance of NRRs in translation.

4 Conclusion

We present a head-driven hierarchical phrase-based (HD-HPB) translation model, which adopts head information (derived through unlabeled dependency analysis) in the definition of non-terminals to better differentiate among translation rules. In ad-

dition, improved and better integrated reordering rules allow better reordering between consecutive non-terminals through exploration of a larger search space in the derivation. Experimental results on Chinese-English translation across four test sets demonstrate significant improvements of the HD-HPB model over both Chiang's HPB and a source-side SAMT-style refined version of HPB.

Acknowledgments

This work was supported by Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University. It was also partially supported by Project 90920004 under the National Natural Science Foundation of China and Project 2012AA011102 under the "863" National High-Tech Research and Development of China. We thank the reviewers for their insightful comments.

References

Hala Almaghout, Jie Jiang, and Andy Way. 2011. CCG contextual labels in hierarchical phrase-based SMT. In *Proceedings of EAMT 2011*, pages 281–288.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of ACL 2007*, pages 33–40.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL 2000*, pages 132–139.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL 2005*, pages 263–270.

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

Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Yang Gao, Philipp Koehn, and Alexandra Birch. 2011. Soft dependency constraints for reordering in hierarchical phrase-based translation. In *Proceedings of EMNLP 2011*, pages 857–868.

Zhongjun He, Yao Meng, and Hao Yu. 2010. Maximum entropy based phrase reordering for hierarchical phrase-based translation. In *Proceedings of EMNLP 2010*, pages 555–563.

Zhongqiang Huang, Martin Cmejrek, and Bowen Zhou. 2010. Soft syntactic constraints for hierarchical phrase-based translation using latent syntactic distributions. In *Proceedings of EMNLP 2010*, pages 138–147.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL 2003*, pages 48–54.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *Proceedings of ACL-HLT 2008*, pages 1003–1011.

Markos Mylonakis and Khalil Sima'an. 2011. Learning hierarchical translation structure with linguistic annotations. In *Proceedings of ACL-HLT 2011*, pages 642–652.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL 2000*, pages 440–447.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

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

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL 2007*, pages 404–411.

Libin Shen, Jinxi Xu, Bing Zhang, Spyros Matsoukas, and Ralph Weischedel. 2009. Effective use of linguistic and contextual information for statistical machine translation. In *Proceedings of EMNLP 2009*, pages 72–80.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of NAACL 2006 - Workshop on Statistical Machine Translation*, pages 138–141.

Andreas Zollmann and Stephan Vogel. 2011. A word-class approach to labeling PSCFG rules for machine translation. In *Proceedings of ACL-HLT 2011*, pages 1–11.

Joint Learning of a Dual SMT System for Paraphrase Generation

Hong Sun[*]
School of Computer Science and Technology
Tianjin University
kaspersky@tju.edu.cn

Ming Zhou
Microsoft Research Asia
mingzhou@microsoft.com

Abstract

SMT has been used in paraphrase generation by translating a source sentence into another (pivot) language and then back into the source. The resulting sentences can be used as candidate paraphrases of the source sentence. Existing work that uses two independently trained SMT systems cannot directly optimize the paraphrase results. Paraphrase criteria especially the paraphrase rate is not able to be ensured in that way. In this paper, we propose a joint learning method of two SMT systems to optimize the process of paraphrase generation. In addition, a revised BLEU score (called $iBLEU$) which measures the adequacy and diversity of the generated paraphrase sentence is proposed for tuning parameters in SMT systems. Our experiments on NIST 2008 testing data with automatic evaluation as well as human judgments suggest that the proposed method is able to enhance the paraphrase quality by adjusting between semantic equivalency and surface dissimilarity.

1 Introduction

Paraphrasing (at word, phrase, and sentence levels) is a procedure for generating alternative expressions with an identical or similar meaning to the original text. Paraphrasing technology has been applied in many NLP applications, such as machine translation (MT), question answering (QA), and natural language generation (NLG).

As paraphrasing can be viewed as a translation process between the original expression (as input) and the paraphrase results (as output), both in the same language, statistical machine translation (SMT) has been used for this task. Quirk et al. (2004) build a monolingual translation system using a corpus of sentence pairs extracted from news articles describing same events. Zhao et al. (2008a) enrich this approach by adding multiple resources (e.g., thesaurus) and further extend the method by generating different paraphrase in different applications (Zhao et al., 2009). Performance of the monolingual MT-based method in paraphrase generation is limited by the large-scale paraphrase corpus it relies on as the corpus is not readily available (Zhao et al., 2010).

In contrast, bilingual parallel data is in abundance and has been used in extracting paraphrase (Bannard and Callison-Burch, 2005; Zhao et al., 2008b; Callison-Burch, 2008; Kok and Brockett, 2010; Kuhn et al., 2010; Ganitkevitch et al., 2011). Thus researchers leverage bilingual parallel data for this task and apply two SMT systems (dual SMT system) to translate the original sentences into another pivot language and then translate them back into the original language. For question expansion, Dubouć and Chu-Carroll (2006) paraphrase the questions with multiple MT engines and select the best paraphrase result considering cosine distance, length, etc. Max (2009) generates paraphrase for a given segment by forcing the segment being translated independently in both of the translation processes. Context features are added into the SMT system to improve translation correctness against polysemous. To reduce

[1]This work has been done while the author was visiting Microsoft Research Asia.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 38–42,
Jeju, Republic of Korea, 8-14 July 2012.

the noise introduced by machine translation, Zhao et al. (2010) propose combining the results of multiple machine translation engines' by performing MBR (Minimum Bayes Risk) (Kumar and Byrne, 2004) decoding on the N-best translation candidates.

The work presented in this paper belongs to the pivot language method for paraphrase generation. Previous work employs two separately trained SMT systems the parameters of which are tuned for SMT scheme and therefore cannot directly optimize the paraphrase purposes, for example, optimize the diversity against the input. Another problem comes from the contradiction between two criteria in paraphrase generation: adequacy measuring the semantic equivalency and paraphrase rate measuring the surface dissimilarity. As they are incompatible (Zhao and Wang, 2010), the question arises how to adapt between them to fit different application scenarios. To address these issues, in this paper, we propose a joint learning method of two SMT systems for paraphrase generation. The jointly-learned dual SMT system: (1) Adapts the SMT systems so that they are tuned specifically for paraphrase generation purposes, e.g., to increase the dissimilarity; (2) Employs a revised BLEU score (named $iBLEU$, as it's an input-aware BLEU metric) that measures adequacy and dissimilarity of the paraphrase results at the same time. We test our method on NIST 2008 testing data. With both automatic and human evaluations, the results show that the proposed method effectively balance between adequacy and dissimilarity.

2 Paraphrasing with a Dual SMT System

We focus on sentence level paraphrasing and leverage homogeneous machine translation systems for this task bi-directionally. Generating sentential paraphrase with the SMT system is done by first translating a source sentence into another pivot language, and then back into the source. Here, we call these two procedures a dual SMT system. Given an English sentence e_s, there could be n candidate translations in another language F, each translation could have m candidates $\{e^{'}\}$ which may contain potential paraphrases for e_s. Our task is to locate the candidate that best fit in the demands of paraphrasing.

2.1 Joint Inference of Dual SMT System

During the translation process, it is needed to select a translation from the hypothesis based on the quality of the candidates. Each candidate's quality can be expressed by log-linear model considering different SMT features such as translation model and language model.

When generating the paraphrase results for each source sentence e_s, the selection of the best paraphrase candidate e'^* from $e' \in C$ is performed by:

$$e'^*(e_s, \{f\}, \lambda^M) = \arg\max_{e' \in C, f \in \{f\}} \sum_{m=1}^{M} \lambda_m h_m(e'|f) t(e', f) \quad (1)$$

where $\{f\}$ is the set of sentences in pivot language translated from e_s, h_m is the m_{th} feature value and λ_m is the corresponding weight. t is an indicator function equals to 1 when e' is translated from f and 0 otherwise.

The parameter weight vector λ is trained by MERT (Minimum Error Rate Training) (Och, 2003). MERT integrates the automatic evaluation metrics into the training process to achieve optimal end-to-end performance. In the joint inference method, the feature vector of each e' comes from two parts: vector of translating e_s to $\{f\}$ and vector of translating $\{f\}$ to e', the two vectors are jointly learned at the same time:

$$(\lambda_1^*, \lambda_2^*) = \arg\max_{(\lambda_1, \lambda_2)} \sum_{s=1}^{S} G(r_s, e'^*(e_s, \{f\}, \lambda_1, \lambda_2)) \quad (2)$$

where G is the automatic evaluation metric for paraphrasing. S is the development set for training the parameters and for each source sentence several human translations r_s are listed as references.

2.2 Paraphrase Evaluation Metrics

The joint inference method with MERT enables the dual SMT system to be optimized towards the quality of paraphrasing results. Different application scenarios of paraphrase have different demands on the paraphrasing results and up to now, the widely mentioned criteria include (Zhao et al., 2009; Zhao et al., 2010; Liu et al., 2010; Chen and Dolan, 2011; Metzler et al., 2011): Semantic adequacy, fluency

and dissimilarity. However, as pointed out by (Chen and Dolan, 2011), there is the lack of automatic metric that is capable to measure all the three criteria in paraphrase generation. Two issues are also raised in (Zhao and Wang, 2010) about using automatic metrics: paraphrase changes less gets larger BLEU score and the evaluations of paraphrase quality and rate tend to be incompatible.

To address the above problems, we propose a metric for tuning parameters and evaluating the quality of each candidate paraphrase c :

$$\begin{aligned} iBLEU(s, r_s, c) &= \alpha BLEU(c, r_s) \\ &- (1-\alpha) BLEU(c, s) \quad (3) \end{aligned}$$

where s is the input sentence, r_s represents the reference paraphrases. $BLEU(c, r_s)$ captures the semantic equivalency between the candidates and the references (Finch et al. (2005) have shown the capability for measuring semantic equivalency using BLEU score); $BLEU(c, s)$ is the BLEU score computed between the candidate and the source sentence to measure the dissimilarity. α is a parameter taking balance between adequacy and dissimilarity, smaller α value indicates larger punishment on self-paraphrase. Fluency is not explicitly presented because there is high correlation between fluency and adequacy (Zhao et al., 2010) and SMT has already taken this into consideration. By using $iBLEU$, we aim at adapting paraphrasing performance to different application needs by adjusting α value.

3 Experiments and Results

3.1 Experiment Setup

For English sentence paraphrasing task, we utilize Chinese as the pivot language, our experiments are built on English and Chinese bi-directional translation. We use 2003 NIST Open Machine Translation Evaluation data (NIST 2003) as development data (containing 919 sentences) for MERT and test the performance on NIST 2008 data set (containing 1357 sentences). NIST Chinese-to-English evaluation data offers four English human translations for every Chinese sentence. For each sentence pair, we choose one English sentence e_1 as source and use the three left sentences e_2, e_3 and e_4 as references.

The English-Chinese and Chinese-English systems are built on bilingual parallel corpus containing 497,862 sentences. Language model is trained on 2,007,955 sentences for Chinese and 8,681,899 sentences for English. We adopt a phrase based MT system of Chiang (2007). 10-best lists are used in both of the translation processes.

Joint learning	BLEU	Self-BLEU	$iBLEU$
No Joint	27.16	35.42	/
$\alpha = 1$	30.75	53.51	30.75
$\alpha = 0.9$	28.28	48.08	20.64
$\alpha = 0.8$	27.39	35.64	14.78
$\alpha = 0.7$	23.27	26.30	8.39

Table 1: $iBLEU$ Score Results(NIST 2008)

	Adequacy (0/1/2)	Fluency (0/1/2)	Variety (0/1/2)	Overall (0/1/2)
No Joint	30/82/88	22/83/95	25/117/58	23/127/50
$\alpha = 1$	33/53/114	15/80/105	62/127/11	16/128/56
$\alpha = 0.9$	31/77/92	16/93/91	23/157/20	20/119/61
$\alpha = 0.8$	31/78/91	19/91/90	20/123/57	19/121/60
$\alpha = 0.7$	35/105/60	32/101/67	9/108/83	35/107/58

Table 2: Human Evaluation Label Distribution

3.2 Paraphrase Evaluation Results

The results of paraphrasing are illustrated in Table 1. We show the BLEU score (computed against references) to measure the adequacy and self-BLEU (computed against source sentence) to evaluate the dissimilarity (lower is better). By "No Joint", it means two independently trained SMT systems are employed in translating sentences from English to Chinese and then back into English. This result is listed to indicate the performance when we do not involve joint learning to control the quality of paraphrase results. For joint learning, results of α from 0.7 to 1 are listed.

From the results we can see that, when the value of α decreases to address more penalty on self-paraphrase, the self-BLEU score rapidly decays while the consequence effect is that BLEU score computed against references also drops seriously. When α drops under 0.6 we observe the sentences become completely incomprehensible (this is the reason why we leave out showing the results of α under 0.7). The best balance is achieved when α is between 0.7 and 0.9, where both of the sentence quality and variety are relatively preserved. As α value is manually defined and not specially tuned, the exper-

Source	Torrential rains hit western india , 43 people dead
No Joint	**Rainstorms in** western india , 43 **deaths**
Joint($\alpha = 1$)	**Rainstorms** hit western india , 43 people dead
Joint($\alpha = 0.9$)	**Rainstorms** hit western india 43 people dead
Joint($\alpha = 0.8$)	**Heavy rain in** western india , 43 **dead**
Joint($\alpha = 0.7$)	**Heavy rain in** western india , 43 **killed**

Table 3: Example of the Paraphrase Results

iments only achieve comparable results with no joint learning when α equals 0.8. However, the results show that our method is able to effectively control the self-paraphrase rate and lower down the score of self-BLEU, this is done by both of the process of joint learning and introducing the metric of $iBLEU$ to avoid trivial self-paraphrase. It is not capable with no joint learning or with the traditional BLEU score does not take self-paraphrase into consideration.

Human evaluation results are shown in Table 2. We randomly choose 100 sentences from testing data. For each setting, two annotators are asked to give scores about semantic adequacy, fluency, variety and overall quality. The scales are 0 (meaning changed; incomprehensible; almost same; cannot be used), 1 (almost same meaning; little flaws; containing different words; may be useful) and 2 (same meaning; good sentence; different sentential form; could be used). The agreements between the annotators on these scores are 0.87, 0.74, 0.79 and 0.69 respectively. From the results we can see that human evaluations are quite consistent with the automatic evaluation, where higher BLEU scores correspond to larger number of good adequacy and fluency labels, and higher self-BLEU results tend to get lower human evaluations over dissimilarity.

In our observation, we found that adequacy and fluency are relatively easy to be kept especially for short sentences. In contrast, dissimilarity is not easy to achieve. This is because the translation tables are used bi-directionally so lots of source sentences' fragments present in the paraphrasing results.

We show an example of the paraphrase results under different settings. All the results' sentential forms are not changed comparing with the input sentence and also well-formed. This is due to the short length of the source sentence. Also, with smaller value of α, more variations show up in the paraphrase results.

4 Discussion

4.1 SMT Systems and Pivot Languages

We have test our method by using homogeneous SMT systems and a single pivot language. As the method highly depends on machine translation, a natural question arises to what is the impact when using different pivots or SMT systems. The joint learning method works by combining both of the processes to concentrate on the final objective so it is not affected by the selection of language or SMT model.

In addition, our method is not limited to a homogeneous SMT model or a single pivot language. As long as the models' translation candidates can be scored with a log-linear model, the joint learning process can tune the parameters at the same time. When dealing with multiple pivot languages or heterogeneous SMT systems, our method will take effect by optimizing parameters from both the forward and backward translation processes, together with the final combination feature vector, to get optimal paraphrase results.

4.2 Effect of $iBLEU$

$iBLEU$ plays a key role in our method. The first part of $iBLEU$, which is the traditional BLEU score, helps to ensure the quality of the machine translation results. Further, it also helps to keep the semantic equivalency. These two roles unify the goals of optimizing translation and paraphrase adequacy in the training process.

Another contribution from $iBLEU$ is its ability to balance between adequacy and dissimilarity as the two aspects in paraphrasing are incompatible (Zhao and Wang, 2010). This is not difficult to explain because when we change many words, the meaning and the sentence quality are hard to preserve. As the paraphrasing task is not self-contained and will be employed by different applications, the two measures should be given different priorities based on the application scenario. For example, for a query

expansion task in QA that requires higher recall, variety should be considered first. Lower α value is preferred but should be kept in a certain range as significant change may lead to the loss of constraints presented in the original sentence. The advantage of the proposed method is reflected in its ability to adapt to different application requirements by adjusting the value of α in a reasonable range.

5 Conclusion

We propose a joint learning method for pivot language-based paraphrase generation. The jointly learned dual SMT system which combines the training processes of two SMT systems in paraphrase generation, enables optimization of the final paraphrase quality. Furthermore, a revised BLEU score that balances between paraphrase adequacy and dissimilarity is proposed in our training process. In the future, we plan to go a step further to see whether we can enhance dissimilarity with penalizing phrase tables used in both of the translation processes.

References

Colin J. Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *ACL*.

Chris Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *EMNLP*, pages 196–205.

David Chen and William B. Dolan. 2011. Collecting highly parallel data for paraphrase evaluation. In *ACL*, pages 190–200.

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

Pablo Ariel Duboué and Jennifer Chu-Carroll. 2006. Answering the question you wish they had asked: The impact of paraphrasing for question answering. In *HLT-NAACL*.

Andrew Finch, Young-Sook Hwang, and Eiichiro Sumita. 2005. Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *In IWP2005*.

Juri Ganitkevitch, Chris Callison-Burch, Courtney Napoles, and Benjamin Van Durme. 2011. Learning sentential paraphrases from bilingual parallel corpora for text-to-text generation. In *EMNLP*, pages 1168–1179.

Stanley Kok and Chris Brockett. 2010. Hitting the right paraphrases in good time. In *HLT-NAACL*, pages 145–153.

Roland Kuhn, Boxing Chen, George F. Foster, and Evan Stratford. 2010. Phrase clustering for smoothing tm probabilities - or, how to extract paraphrases from phrase tables. In *COLING*, pages 608–616.

Shankar Kumar and William J. Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *HLT-NAACL*, pages 169–176.

Chang Liu, Daniel Dahlmeier, and Hwee Tou Ng. 2010. Pem: A paraphrase evaluation metric exploiting parallel texts. In *EMNLP*, pages 923–932.

Aurelien Max. 2009. Sub-sentential paraphrasing by contextual pivot translation. In *Proceedings of the 2009 Workshop on Applied Textual Inference, ACLI-JCNLP*, pages 18–26.

Donald Metzler, Eduard H. Hovy, and Chunliang Zhang. 2011. An empirical evaluation of data-driven paraphrase generation techniques. In *ACL (Short Papers)*, pages 546–551.

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

Chris Quirk, Chris Brockett, and William B. Dolan. 2004. Monolingual machine translation for paraphrase generation. In *EMNLP*, pages 142–149.

Shiqi Zhao and Haifeng Wang. 2010. Paraphrases and applications. In *COLING (Tutorials)*, pages 1–87.

Shiqi Zhao, Cheng Niu, Ming Zhou, Ting Liu, and Sheng Li. 2008a. Combining multiple resources to improve smt-based paraphrasing model. In *ACL*, pages 1021–1029.

Shiqi Zhao, Haifeng Wang, Ting Liu, and Sheng Li. 2008b. Pivot approach for extracting paraphrase patterns from bilingual corpora. In *ACL*, pages 780–788.

Shiqi Zhao, Xiang Lan, Ting Liu, and Sheng Li. 2009. Application-driven statistical paraphrase generation. In *ACL/AFNLP*, pages 834–842.

Shiqi Zhao, Haifeng Wang, Xiang Lan, and Ting Liu. 2010. Leveraging multiple mt engines for paraphrase generation. In *COLING*, pages 1326–1334.

A Novel Burst-based Text Representation Model for Scalable Event Detection

Wayne Xin Zhao[†], **Rishan Chen**[†], **Kai Fan**[†], **Hongfei Yan**[†*] **and Xiaoming Li**[†‡]
[†]School of Electronics Engineering and Computer Science, Peking University, China
[‡]State Key Laboratory of Software, Beihang University, China
{batmanfly,tsunamicrs,fankaicn,yhf1029}@gmail.com, lxm@pku.edu.cn

Abstract

Mining retrospective events from text streams has been an important research topic. Classic text representation model (i.e., vector space model) cannot model temporal aspects of documents. To address it, we proposed a novel burst-based text representation model, denoted as BurstVSM. BurstVSM corresponds dimensions to bursty features instead of terms, which can capture semantic and temporal information. Meanwhile, it significantly reduces the number of non-zero entries in the representation. We test it via scalable event detection, and experiments in a 10-year news archive show that our methods are both effective and efficient.

1 Introduction

Mining retrospective events (Yang et al., 1998; Fung et al., 2007; Allan et al., 2000) has been quite an important research topic in text mining. One standard way for that is to cluster news articles as events by following a two-step approach (Yang et al., 1998): 1) represent document as vectors and calculate similarities between documents; 2) run the clustering algorithm to obtain document clusters as events.[1] Underlying text representation often plays a critical role in this approach, especially for long text streams. In this paper, our focus is to study how to represent temporal documents effectively for event detection.

Classical text representation methods, i.e., Vector Space Model (VSM), have a few shortcomings when dealing with temporal documents. The major one is that it maps one dimension to one term, which completely ignores temporal information, and therefore VSM can never capture the evolving trends in text streams. See the example in Figure 1, D_1 and D_2 may have a high similarity based on VSM due to the presence of some general terms (e.g., "election") related to U.S. presidential election, although general terms correspond to events in different periods (i.e., November 2004 and November 2008). Temporal information has to be taken into consideration for event detection. Another important issue is scalability, with the increasing of the number in the text stream, the size of the vocabulary, i.e., the number of dimensions in VSM, can be very large, which requires a considerable amount of space for storage and time for downstream processing.

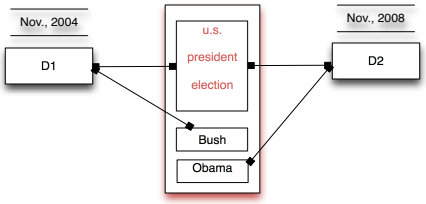


Figure 1: A motivating example. D_1 and D_2 are news articles about U.S. presidential election respectively in years 2004 and 2008.

To address these difficulties, in this paper, we propose a burst based text representation method for scalable event detection. The major novelty is to naturally incorporate temporal information into dimensions themselves instead of using external time decaying functions (Yang et al., 1998). We instantiate this idea by using bursty features as basic representation units of documents. In this paper, *bursty feature* refers to a sudden surge of the frequency of a single term in a text stream, and it is represented as the term itself together with the time interval during which the burst takes place. For example, `(Olympic, Aug-08-2008, Aug-24-2008)`[2] can be regarded as a bursty feature. We also call the term in a bursty

[*]Corresponding author.

[1]Post-processing may be also needed on the preliminary document clusters to refine the results.

[2]Beijing 2008 Olympic Games

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 43–47,
Jeju, Republic of Korea, 8-14 July 2012.

feature its bursty term. In our model, each dimension corresponds to a bursty feature, which contains both temporal and semantic information. Bursty features capture and reflect the evolving topic trends, which can be learnt by searching surge patterns in stream data (Kleinberg, 2003). Built on bursty features, our representation model can well adapt to text streams with complex trends, and therefore provides a more reasonable temporal document representation. We further propose a *split-cluster-merge* algorithm to generate clusters as events. This algorithm can run a mutli-thread mode to speed up processing.

Our contribution can be summarized as two aspects: 1) we propose a novel burst-based text representation model, to our best knowledge, it is the first work which explicitly incorporates temporal information into dimensions themselves; 2) we test this representation model via scalable event detection task on a very large news corpus, and extensive experiments show the proposed methods are both effective and efficient.

2 Burst-based Text Representation

In this section, we describe the proposed burst-based text representation model, denoted as *BurstVSM*. In BurstVSM, each document is represented as one vector as in VSM, while the major novelty is that one dimension is mapped to one bursty feature instead of one term. In this paper, we define a bursty feature f as a triplet (w^f, t_s^f, t_e^f), where w is the bursty term and t_s and t_e are the start and end timestamps of the bursty interval (period). Before introducting BurstVSM, we first discuss how to identify bursty features from text streams.

2.1 Burst Detection Algorithm

We follow the batch mode two-state automaton method from (Kleinberg, 2003) for bursty feature detection.[3] In this model, a stream of documents containing a term w are assumed to be generated from a two-state automaton with a low frequency state q_0 and a high frequency state q_1. Each state has its own emission rate (p_0 and p_1 respectively), and there is a probability for changing state. If an interval of high states appears in the optimal state sequence of some term, this term together with this interval is detected as a bursty feature. To obtain all bursty features in text streams, we can perform burst detection on each term in the vocabulary. Instead of using a fixed p_0 and p_1 in (Kleinberg, 2003), by following the moving average method (Vlachos et al., 2004) ,we parameterize p_0 and p_1 with the time index for each batch, formally, we have $p_0(t)$ and $p_1(t)$ for the tth batch. Given a term w, we use a sliding window of length L to estimate $p_0(t)$ and $p_1(t)$ for the tth batch as follows: $p_0(t) = \frac{\sum_{j \in W_t} N_{j,w}}{\sum_{j \in W_t} N_j}$ and $p_1(t) = p_0(t) \times s$, where $N_{j,w}$ and N_j are w 's document frequency and the total number of documents in jth batch respectively. s is a scaling factor lager than 1.0, indicating state q_1 has a faster rate, and it is empirically set as 1.5. W_t is a time interval $[\max(t - L/2, 0),\ \min(t + L/2, N)]$, and the length of moving window L is set as 180 days. All the other parts remain the same as in (Kleinberg, 2003). Our detection method is denoted as *TVBurst*.

[3]The news articles in one day is treated as a batch.

2.2 Burst based text representation models

We apply TVBurst to all the terms in our vocabulary to identify a set of bursty features, denoted as $\mathcal{B}$. Given $\mathcal{B}$, a document $\mathbf{d}_i(t)$ with timestamp t is represented as a vector of weights in ***bursty feature dimensions***:

$$\mathbf{d}_i(t) = (d_{i,1}(t), d_{i,2}(t), ..., d_{i,|\mathcal{B}|}(t)).$$

We define the jth weight of $\mathbf{d}_i$ as follows

$$d_{i,j} = \begin{cases} \text{tf-idf}_{i,w^{\mathcal{B}_j}}, & \text{if } t \in [t_s^{\mathcal{B}_j}, t_e^{\mathcal{B}_j}], \\ 0, & \text{otherwise.} \end{cases}$$

When the timestamp of $\mathbf{d}_i$ is in the bursty interval of $\mathcal{B}_j$ and contains bursty term $w^{\mathcal{B}_j}$, we set up the weight using common used *tf-idf* method. In BurstVSM, each dimension is mapped to one bursty feature, and it considers both semantic and temporal information. One dimension is active only when the document falls in the corresponding bursty interval. Usually, a document vector in BurstVSM has only a few non-zero entries, which makes computation of document similarities more efficient in large datasets compared with traditional VSM.

The most related work to ours is the boostVSM introduced by (He et al., 2007b), it proposes to weight different term dimensions with corresponding bursty scores. However, it is still based on term dimensions and fails to deal with terms with multiple bursts. Suppose that we are dealing with a text collection related with U.S. presidential elections, Fig. 2 show sample dimensions for these three methods. In BurstVSM, one term with multiple bursts will be naturally mapped to different dimensions. For example, two bursty features (presidential, Nov., 2004) and (presidential, Nov., 2008) correspond to different dimensions in BurstVSM, while

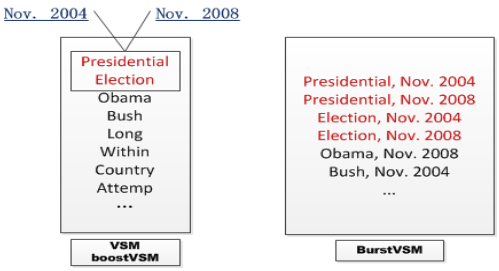


Figure 2: One example for comparisons of different representation methods. Terms in red box correspond to multiple bursty periods.

Table 1: Summary of different representation models. Here dimension reduction refers to the reduction of non-zero entries in representation vector.

	semantic information	temporal information	dimension reduction	trend modeling
VSM	✓	×	×	bad
boostVSM	✓	partially	×	moderate
BurstVSM	✓	✓	✓	good

VSM and boostVSM cannot capture such temporal differences. Some methods try to design time decaying functions (Yang et al., 1998), which decay the similarity with the increasing of time gap between two documents. However, it requires efforts for function selection and parameters tuning. We summarize these discussions in Table 1.

3 *split-cluster-merge* algorithm for event detection

In this section, we discuss how to cluster documents as events. Since each document can be represented as a burst-based vector, we use cosine function to compute document similarities. Due to the large size of our news corpus, it is infeasible to cluster all the documents straightforward. We develop a heuristic clustering algorithm for event detection, denoted as *split-cluster-merge*, which includes three main steps, namely split, cluster and merge. The idea is that we first split the dataset into small parts, then cluster the documents of each part independently and finally merge similar clusters from two consecutive parts. In our dataset, we find that most events last no more than one month, so we split the dataset into parts by months. After splitting, clustering can run in parallel for different parts (we use *CLUTO*[4] as the clustering tool), which significantly reduces total time cost. For *merge*, we merge clusters in consecutive months with an empirical threshold of 0.5. The final clusters are returned as identified events.

4 Evaluation

4.1 Experiment Setup

We used a subset of 68 millon deduplicated timestamped web pages generated from this archive (Huang et al., 2008). Since our major focus is to detect events from news articles, we only keep the web pages with keyword "news" in *URL* field. The final collection contains $11,218,581$ articles with total $1,730,984,304$ tokens ranging from 2000 to 2009. We run all the experiments on a 64-bit linux server with four Quad-Core AMD Opteron(tm) Processors and 64GB of RAM. For *split-cluster-merge* algorithm, we implement the *cluster* step in a multi-thread mode, so that different parts can be processed in parallel.

4.2 Construction of test collection

We manually construct the test collection for event detection. To examine the effectiveness of event detection methods in different grains, we consider two type of events in terms of the number of relevant documents, namely significant events and moderate events. A significant event is required to have at least 300 relevant docs, and a moderate event is required to have $10 \sim 100$ relevant docs. 14 graduate students are invited to generate the test collection, starting with a list of 100 candidate seed events by referring to Xinhua News.[5] For one target event, the judges first construct queries with temporal constraints to retrieve candidate documents and then judge wether they are relevant or not. Each document is assigned to three students, and we adopt the majority-win strategy for the final judgment. Finally, by removing all candidate seed events which neither belong to significant events nor moderate events, we derive a test collection consisting of 24 significant events and 40 moderate events.[6]

4.3 Evaluation metrics and baselines

Similar to the evaluation in information retrieval , given a target event, we evaluate the quality of the returned "relevant" documents by systems. We use average precision, average recall and mean average precision(MAP) as evaluation metrics. A difference is that we do not have queries, and the output of a system is a set of document clusters. So for a system, given an event in golden standard, we first select the cluster (the system generates) which has the

[4] www.cs.umn.edu/~karypis/cluto

[5] http://news.xinhuanet.com/english

[6] For access to the code and test collection, contact Xin Zhao via batmanfly@gmail.com.

Table 2: Results of event detection. Our proposed method is better than all the other baselines at confidence level 0.9.

	Signifcant Events				**Moderate Events**			
	P	*R*	*F*	*MAP*	*P*	*R*	*F*	*MAP*
timemines-χ^2(nouns)	0.52	0.2	0.29	0.11	0.22	0.27	0.24	0.09
timemines-χ^2(NE)	0.61	0.18	0.28	0.08	0.27	0.25	0.26	0.13
TVBurst+boostVSM	0.67	0.44	0.53	0.31	0.22	0.39	0.28	0.13
swan+BurstVSM	0.74	0.56	0.64	0.48	0.39	0.54	0.45	0.38
kleiberg+BurstVSM	0.68	0.63	0.65	0.52	0.35	0.53	0.42	0.36
TVBurst+BurstVSM	**0.78**	**0.69**	**0.73**	**0.63**	**0.4**	**0.61**	**0.48**	**0.39**

Table 3: Comparisons of average intra-class and inter-class similarity.

Methods	**Significant Events**		**Moderate Events**	
	Intra	Inter	Intra	Inter
TVBurst+boostVSM	0.234	0.132	0.295	0.007
TVBurst+BurstVSM	0.328	0.014	0.480	0.004

Table 4: Comparisons of observed runtime and storage.

	boostVSM	**BurstVSM**
Aver. # of non-zero entries per doc	149	14
File size for storing vectors (gigabytes)	3.74	0.571
Total # of *merge*	10,265,335	9,801,962
Aver. *cluster* cost per month (sec.)	355	55
Total *merge* cost (sec.)	2,441	875
Total time cost (sec.)	192,051	4,851

most relevant documents, then sort the documents in the descending order of similarities with the cluster centroid and finally compute P, R ,F and MAP in this cluster. We perform Wilcoxon signed-rank test for significance testing.

We used the event detection method in (Swan and Allan, 2000) as baseline, denoted as timemines-χ^2. As (Swan and Allan, 2000) suggested, we tried two versions: 1) using all nouns and 2) using all named entities. Recall that BurstVSM relies on bursty features as dimensions, we tested different burst detection algorithms in our proposed BurstVSM model, including *swan* (Swan and Allan, 2000), *kleinberg* (Kleinberg, 2003) and our proposed *TVBurst* algorithm.

4.4 Experiment results

Preliminary results. In Table 2, we can see that 1) *BurstVSM* with any of these three burst detection algorithms is significantly better than timemines-χ^2, suggesting our event detection method is very effective; 2) *TVBurst* with *BurstVSM* gives the best performance, which suggests using moving average base probability will improve the performance of burst detection. We use *TVBurst* as the default burst detection algorithm in later experiments.

Then we compare the performance of different text representation models for event detection, namely *BurstVSM* and *boostVSM* (He et al., 2007b; He et al., 2007a).[7] For different representation models, we use *split-cluster-merge* as clustering algorithm. Table 2 shows that BurstVSM is much effecitve than boostVSM for event detection. In fact, we empirically find boostVSM is appropriate for clustering documents in a coarse grain (e.g., in topic level) but not for event detection.

Intra-class and inter-class similarities. In our methods, event detection is treated as document clustering. It is very important to study how similarities affect the performance of clustering. To see why our proposed representation methods are better than *boostVSM*, we present the average intra-class similarity and inter-class similarity for different events in Table 3.[8] We can see BurstVSM results in a larger intra-class similarity and a smaller inter-class similarity than boostVSM.

Analysis of the space/time complexity. We further analyze the space/time complexity of different representation models. In Table 4. We can see that BurstVSM has much smaller space/time cost compared with boostVSM, and meanwhile it has a better performance for event detection (See Table 2). In burst-based representation, one document has fewer non-zero entries.

Acknowledgement. The core idea of this work is initialized and developped by Kai Fan. This work is partially supported by HGJ 2010 Grant 2011ZX01042-001-001, NSFC Grant 61073082 and 60933004. Xin Zhao is supported by Google PhD Fellowship (China). We thank the insightful comments from Junjie Yao, Jing Liu and the anonymous reviewers. We have developped an online Chinese large-scale event search engine based on this work, visit http://sewm.pku.edu.cn/eventsearch for more details.

[7] We use the same parameter settings in the original paper.

[8] For each event in our golden standard, we have two clusters: relevant documents and non-relevant documents(within the event period).

References

James Allan, Victor Lavrenko, and Hubert Jin. 2000. First story detection in TDT is hard. In *Proceedings of the ninth international conference on Information and knowledge management*.

Gabriel Pui Cheong Fung, Jeffrey Xu Yu, Huan Liu, and Philip S. Yu. 2007. Time-dependent event hierarchy construction. In *SIGKDD*.

Q. He, K. Chang, and E. P. Lim. 2007a. Using burstiness to improve clustering of topics in news streams. In *ICDM*.

Qi He, Kuiyu Chang, Ee-Peng Lim, and Jun Zhang. 2007b. Bursty feature representation for clustering text streams. In *SDM*.

L. Huang, L. Wang, and X. Li. 2008. Achieving both high precision and high recall in near-duplicate detection. In *CIKM*.

J. Kleinberg. 2003. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*.

Russell Swan and James Allan. 2000. Automatic generation of overview timelines. In *SIGIR*.

Michail Vlachos, Christopher Meek, Zografoula Vagena, and Dimitrios Gunopulos. 2004. Identifying similarities, periodicities and bursts for online search queries. In *SIGMOD*.

Yiming Yang, Tom Pierce, and Jaime Carbonell. 1998. A study of retrospective and on-line event detection. In *SIGIR*.

A Graph-based Cross-lingual Projection Approach for Weakly Supervised Relation Extraction

Seokhwan Kim
Human Language Technology Dept.
Institute for Infocomm Research
Singapore 138632
kims@i2r.a-star.edu.sg

Gary Geunbae Lee
Dept. of Computer Science and Engineering
Pohang University of Science and Technology
Pohang, 790-784, Korea
gblee@postech.ac.kr

Abstract

Although researchers have conducted extensive studies on relation extraction in the last decade, supervised approaches are still limited because they require large amounts of training data to achieve high performances. To build a relation extractor without significant annotation effort, we can exploit cross-lingual annotation projection, which leverages parallel corpora as external resources for supervision. This paper proposes a novel graph-based projection approach and demonstrates the merits of it by using a Korean relation extraction system based on projected dataset from an English-Korean parallel corpus.

1 Introduction

Relation extraction aims to identify semantic relations of entities in a document. Although many supervised machine learning approaches have been successfully applied to relation extraction tasks (Zelenko et al., 2003; Kambhatla, 2004; Bunescu and Mooney, 2005; Zhang et al., 2006), applications of these approaches are still limited because they require a sufficient number of training examples to obtain good extraction results. Several datasets that provide manual annotations of semantic relationships are available from MUC (Grishman and Sundheim, 1996) and ACE (Doddington et al., 2004) projects, but these datasets contain labeled training examples in only a few major languages, including English, Chinese, and Arabic. Although these datasets encourage the development of relation extractors for these major languages, there are few labeled training samples for learning new systems in other languages, such as Korean. Because manual annotation of semantic relations for such *resource-poor languages* is very expensive, we instead consider weakly supervised learning techniques (Riloff and Jones, 1999; Agichtein and Gravano, 2000; Zhang, 2004; Chen et al., 2006) to learn the relation extractors without significant annotation efforts. But these techniques still face cost problems when preparing quality seed examples, which plays a crucial role in obtaining good extractions.

Recently, some researchers attempted to use external resources, such as treebank (Banko et al., 2007) and Wikipedia (Wu and Weld, 2010), that were not specially constructed for relation extraction instead of using task-specific training or seed examples. We previously proposed to leverage parallel corpora as a new kind of external resource for relation extraction (Kim et al., 2010). To obtain training examples in the resource-poor target language, this approach exploited a *cross-lingual annotation projection* by propagating annotations that were generated by a relation extraction system in a resource-rich source language. In this approach, projected annotations were determined in a single pass process by considering only alignments between entity candidates; we call this action *direct projection*.

In this paper, we propose a graph-based projection approach for weakly supervised relation extraction. This approach utilizes a graph that is constructed with both instance and context information and that is operated in an iterative manner. The goal of our graph-based approach is to improve the robustness of the extractor with respect to errors that are generated and accumulated by preprocessors.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 48–53,
Jeju, Republic of Korea, 8-14 July 2012.

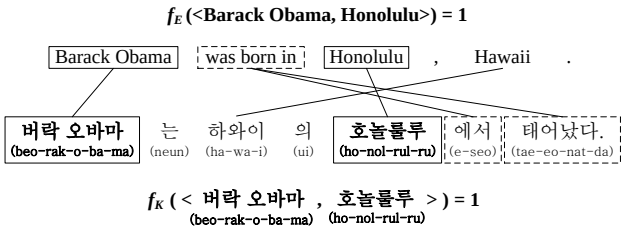


Figure 1: An example of annotation projection for relation extraction of a bitext in English and Korean

2 Cross-lingual Annotation Projection for Relation Extraction

Relation extraction can be considered to be a classification problem by the following classifier:

$$f\left(e^i, e^j\right) = \begin{cases} 1 & \text{if } e^i \text{ and } e^j \text{ have a relation,} \\ -1 & \text{otherwise.} \end{cases},$$

where e_i and e_j are entities in a sentence.

Cross-lingual annotation projection intends to learn an extractor f_t for good performance without significant effort toward building resources for a resource-poor target language L_t. To accomplish that goal, the method automatically creates a set of annotated text for f_t, utilizing a well-made extractor f_s for a resource-rich source language L_s and a parallel corpus of L_s and L_t. Figure 1 shows an example of annotation projection for relation extraction with a bi-text in L_t Korean and L_s English. Given an English sentence, an instance ⟨*Barack Obama, Honolulu*⟩ is extracted as positive. Then, its translational counterpart ⟨*beo-rak-o-ba-ma, ho-nol-rul-ru*⟩ in the Korean sentence also has a positive annotation by projection.

Early studies in cross-lingual annotation projection were accomplished for various natural language processing tasks (Yarowsky and Ngai, 2001; Yarowsky et al., 2001; Hwa et al., 2005; Zitouni and Florian, 2008; Pado and Lapata, 2009). These studies adopted a simple direct projection strategy that propagates the annotations in the source language sentences to word-aligned target sentences, and a target system can bootstrap from these projected annotations.

For relation extraction, the direct projection strategy can be formularized as follows: $f_t\left(e_t^i, e_t^j\right) = f_s\left(A(e_t^i), A(e_t^j)\right)$, where $A(e_t)$ is the aligned entity of e_t. However, these automatic annotations can be unreliable because of source text mis-classification and word alignment errors; thus, it can cause a critical falling-off in the annotation projection quality.

Although some noise reduction strategies for projecting semantic relations were proposed (Kim et al., 2010), the direct projection approach is still vulnerable to erroneous inputs generated by submodules. We note two main causes for this limitation: (1) the direct projection approach considers only alignments between entity candidates, and it does not consider any contextual information; and, (2) it is performed by a single pass process. To solve both of these problems at once, we propose a graph-based projection approach for relation extraction.

3 Graph Construction

The most crucial factor in the success of graph-based learning approaches is how to construct a graph that is appropriate for the target task. Das and Petrov (Das and Petrov, 2011) proposed a graph-based bilingual projection of part-of-speech tagging by considering the tagged words in the source language as labeled examples and connecting them to the unlabeled words in the target language, while referring to the word alignments. Graph construction for projecting semantic relationships is more complicated than part-of-speech tagging because the unit instance of projection is a pair of entities and not a word or morpheme that is equivalent to the alignment unit.

3.1 Graph Vertices

To construct a graph for a relation projection, we define two types of vertices: instance vertices V and context vertices U.

Instance vertices are defined for all pairs of entity candidates in the source and target languages. Each instance vertex has a soft label vector $Y = [\ y^+ \quad y^-\ ]$, which contains the probabilities that the instance is positive or negative, respectively. The larger the y^+ value, the more likely the instance has a semantic relationship. The initial label values of an instance vertex $v_s^{ij} \in V_s$ for the instance $\left\langle e_s^i, e_s^j \right\rangle$ in the source language are assigned based on the confidence score of the extractor f_s. With respect to the target language, every instance vertex $v_t^{ij} \in V_t$ has

the same initial values of 0.5 in both y^+ and y^-.

The other type of vertices, context vertices, are used for identifying relation descriptors that are contextual subtexts that represent semantic relationships of the positive instances. Because the characteristics of these descriptive contexts vary depending on the language, context vertices should be defined to be language-specific. In the case of English, we define the context vertex for each trigram that is located between a given entity pair that is semantically related. If the context vertices U_s for the source language sentences are defined, then the units of context in the target language can also be created based on the word alignments. The aligned counterpart of each source language context vertex is used for generating a context vertex $u_t^i \in U_t$ in the target language. Each context vertex $u_s \in U_s$ and $u_t \in U_t$ also has y^+ and y^-, which represent how likely the context is to denote semantic relationships. The probability values for all of the context vertices in both of the languages are initially assigned to $y^+ = y^- = 0.5$.

3.2 Edge Weights

The graph for our graph-based projection is constructed by connecting related vertex pairs by weighted edges. If a given pair of vertices is likely to have the same label, then the edge connecting these vertices should have a large weight value.

We define three types of edges according to combinations of connected vertices. The first type of edges consists of connections between an instance vertex and a context vertex in the same language. For a pair of an instance vertex $v^{i,j}$ and a context vertex u^k, these vertices are connected if the context sequence of $v^{i,j}$ contains u^k as a subsequence. If v^{ij} is matched to u^k, the edge weight $w\left(v^{i,j}, u^k\right)$ is assigned to 1. Otherwise, it should be 0.

Another edge category is for the pairs of context vertices in a language. Because each context vertex is considered to be an n-gram pattern in our work, the weight value for each edge of this type represents the pattern similarity between two context vertices. The edge weight $w(u^k, u^l)$ is computed by Jaccard's coefficient between u^k and u^l.

While the previous two categories of edges are concerned with monolingual connections, the other type addresses bilingual alignments of context vertices between the source language and the target language. We define the weight for a bilingual edge connecting u_s^k and u_t^l as the relative frequency of alignments, as follows:

$$w(u_s^k, u_t^l) = \text{count}\left(u_s^k, u_t^l\right) / \sum_{u_t^m} \text{count}\left(u_s^k, u_t^m\right),$$

where $\text{count}\,(u_s, u_t)$ is the number of alignments between u_s and u_t across the whole parallel corpus.

4 Label Propagation

To induce labels for all of the unlabeled vertices on the graph constructed in Section 3, we utilize the label propagation algorithm (Zhu and Ghahramani, 2002), which is a graph-based semi-supervised learning algorithm.

First, we construct an $n \times n$ matrix T that represents transition probabilities for all of the vertex pairs. After assigning all of the values on the matrix, we normalize the matrix for each row, to make the element values be probabilities. The other input to the algorithm is an $n \times 2$ matrix Y, which indicates the probabilities of whether a given vertex v_i is positive or not. The matrix T and Y are initialized by the values described in Section 3.

For the input matrices T and Y, label propagation is performed by multiplying the two matrices, to update the Y matrix. This multiplication is repeated until Y converges or until the number of iterations exceeds a specific number. The Y matrix, after finishing its iterations, is considered to be the result of the algorithm.

5 Implementation

To demonstrate the effectiveness of the graph-based projection approach for relation extraction, we developed a Korean relation extraction system that was trained with projected annotations from English resources. We used an English-Korean parallel corpus [1] that contains 266,892 bi-sentence pairs in English and Korean. We obtained 155,409 positive instances from the English sentences using an off-the-shelf relation extraction system, ReVerb [2] (Fader et al., 2011).

[1] The parallel corpus collected is available in our website: http://isoft.postech.ac.kr/~megaup/acl/datasets

[2] http://reverb.cs.washington.edu/

Table 1: Comparison between direct and graph-based projection approaches to extract semantic relationships for four relation types

Type	Direct			Graph-based		
	P	R	F	P	R	F
Acquisition	51.6	87.7	64.9	55.3	91.2	68.9
Birthplace	69.8	84.5	76.4	73.8	87.3	80.0
Inventor Of	62.4	85.3	72.1	66.3	89.7	76.3
Won Prize	73.3	80.5	76.7	76.4	82.9	79.5
Total	63.9	84.2	72.7	67.7	87.4	76.3

The English sentence annotations in the parallel corpus were then propagated into the corresponding Korean sentences. We used the GIZA++ software [3] (Och and Ney, 2003) to obtain the word alignments for each bi-sentence in the parallel corpus. The graph-based projection was performed by the Junto toolkit [4] with the maximum number of iterations of 10 for each execution.

Projected instances were utilized as training examples to learn the Korean relation extractor. We built a tree kernel-based support vector machine model using SVM-Light [5] (Joachims, 1998) and Tree Kernel tools [6] (Moschitti, 2006). In our model, we adopted the subtree kernel method for the shortest path dependency kernel (Bunescu and Mooney, 2005).

6 Evaluation

The experiments were performed on the manually annotated Korean test dataset. The dataset was built following the approach of Bunescu and Mooney (Bunescu and Mooney, 2007). The dataset consists of 500 sentences for four relation types: Acquisition, Birthplace, Inventor of, and Won Prize. Of these, 278 sentences were annotated as positive instances.

The first experiment aimed to compare two systems constructed by the direct projection (Kim et al., 2010) and graph-based projection approach. Table 1 shows the performances of the relation extraction of the two systems. The graph-based system achieved better performances in precision and recall than the system with direct projection for all of the four relation types. It outperformed the baseline system by an F-measure of 3.63.

Table 2: Comparisons of our projection approach to heuristic and Wikipedia-based approaches

Approach	P	R	F
Heuristic-based	92.31	17.27	29.09
Wikipedia-based	66.67	66.91	66.79
Projection-based	67.69	87.41	76.30

To demonstrate the merits of our work against other approaches based on monolingual external resources, we performed comparisons with the following two baselines: heuristic-based (Banko et al., 2007) and Wikipedia-based approaches (Wu and Weld, 2010). The heuristic-based baseline was built on the Sejong treebank corpus (Kim, 2006) and the Wikipedia-based baseline used Korean Wikipedia articles [7]. Table 2 compares the performances of the two baseline systems and our method. Our proposed projection-based approach obtained better performance than the other systems. It outperformed the heuristic-based system by 47.21 and the Wikipedia-based system by 9.51 in the F-measure.

7 Conclusions

This paper presented a novel graph-based projection approach for relation extraction. Our approach performed a label propagation algorithm on a proposed graph that represented the instance and context features of both the source and target languages. The feasibility of our approach was demonstrated by our Korean relation extraction system. Experimental results show that our graph-based projection helped to improve the performance of the cross-lingual annotation projection of the semantic relations, and our system outperforms the other systems, which incorporate monolingual external resources.

In this work, we operated the graph-based projection under very restricted conditions, because of high complexity of the algorithm. For future work, we plan to relieve the complexity problem for dealing with more expanded graph structure to improve the performance of our proposed approach.

[3] http://code.google.com/p/giza-pp/

[4] http://code.google.com/p/junto/

[5] http://svmlight.joachims.org/

[6] http://disi.unitn.it/ moschitt/Tree-Kernel.htm

[7] We used the Korean Wikipedia database dump as of June 2011.

Acknowledgments

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program (NIPA-2012-(H0301-12-3001)) supervised by the NIPA(National IT Industry Promotion Agency) and Industrial Strategic technology development program, 10035252, development of dialog-based spontaneous speech interface technology on mobile platform, funded by the Ministry of Knowledge Economy(MKE, Korea).

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 85–94.

M. Banko, M. J Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 2670–2676.

R. Bunescu and R. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 724–731.

R. Bunescu and R. Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th annual meeting of the Association for Computational Linguistics*, volume 45, pages 576–583.

J. Chen, D. Ji, C. L Tan, and Z. Niu. 2006. Relation extraction using label propagation based semi-supervised learning. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 129–136.

D. Das and S. 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.

G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. 2004. The automatic content extraction (ACE) program–tasks, data, and evaluation. In *Proceedings of LREC*, volume 4, pages 837–840.

A. Fader, S. Soderland, and O. Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545.

R. Grishman and B. Sundheim. 1996. Message understanding conference-6: A brief history. In *Proceedings of the 16th conference on Computational linguistics*, volume 1, pages 466–471.

R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural language engineering*, 11(3):311–325.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, pages 137–142.

N. Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, pages 22–25.

S. Kim, M. Jeong, J. Lee, and G. G Lee. 2010. A cross-lingual annotation projection approach for relation detection. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 564–571.

H. Kim. 2006. Korean national corpus in the 21st century sejong project. In *Proceedings of the 13th NIJL International Symposium*, pages 49–54.

A. Moschitti. 2006. Making tree kernels practical for natural language learning. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, volume 6, pages 113–120.

F. J Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

S. Pado and M. Lapata. 2009. Cross-lingual annotation projection of semantic roles. *Journal of Artificial Intelligence Research*, 36(1):307–340.

E. Riloff and R. Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the National Conference on Artificial Intelligence*, pages 474–479.

F. Wu and D. Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127.

D. Yarowsky and G. Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 1–8.

D. Yarowsky, G. Ngai, and R. Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the*

First International Conference on Human Language Technology Research, pages 1–8.

D. Zelenko, C. Aone, and A. Richardella. 2003. Kernel methods for relation extraction. *The Journal of Machine Learning Research*, 3:1083–1106.

M. Zhang, J. Zhang, J. Su, and G. Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 825–832.

Z. Zhang. 2004. Weakly-supervised relation classification for information extraction. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 581–588.

X. Zhu and Z. Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation. *School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-CALD-02-107*.

I. Zitouni and R. Florian. 2008. Mention detection crossing the language barrier. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 600–609.

Pattern Learning for Relation Extraction with a Hierarchical Topic Model

Enrique Alfonseca **Katja Filippova** **Jean-Yves Delort**
Google Research
Brandschenkestrasse 110
8002 Zurich, Switzerland
{ealfonseca,katjaf,jydelort}@google.com

Guillermo Garrido*
NLP & IR Group, UNED
Juan del Rosal, 16.
28040 Madrid, Spain
ggarrido@lsi.uned.es

Abstract

We describe the use of a hierarchical topic model for automatically identifying syntactic and lexical patterns that explicitly state ontological relations. We leverage distant supervision using relations from the knowledge base FreeBase, but do not require any manual heuristic nor manual seed list selections. Results show that the learned patterns can be used to extract new relations with good precision.

1 Introduction

The detection of relations between entities for the automatic population of knowledge bases is very useful for solving tasks such as Entity Disambiguation, Information Retrieval and Question Answering. The availability of high-coverage, general-purpose knowledge bases enable the automatic identification and disambiguation of entities in text and its applications (Bunescu and Pasca, 2006; Cucerzan, 2007; McNamee and Dang, 2009; Kwok et al., 2001; Pasca et al., 2006; Weld et al., 2008; Pereira et al., 2009; Kasneci et al., 2009).

Most early works in this area were designed for supervised Information Extraction competitions such as MUC (Sundheim and Chinchor, 1993) and ACE (ACE, 2004; Doddington et al., 2004; Li et al., 2011), which rely on the availability of annotated data. Open Information Extraction (Sekine, 2006; Banko et al., 2007; Bollegala et al., 2010) started as an effort to approach relation extraction in a completely unsupervised way, by learning regularities and patterns from the web. Two example systems implementing this paradigm are TEXTRUNNER (Yates et al., 2007) and REVERB (Fader et al., 2011). These systems do not need any manual data or rules, but the relational facts they extract are not immediately disambiguated to entities and relations from a knowledge base.

A different family of unsupervised methods for relation extraction is *unsupervised semantic parsing*, which aims at clustering entity mentions and relation surface forms, thus generating a semantic representation of the texts on which inference may be used. Some techniques that have been used are Markov Random Fields (Poon and Domingos, 2009) and Bayesian generative models (Titov and Klementiev, 2011). These are quite powerful approaches but have very high computational requirements (cf. (Yao et al., 2011)).

A good trade-off between fully supervised and fully unsupervised approaches is *distant supervision*, a semi-supervised procedure consisting of finding sentences that contain two entities whose relation we know, and using those sentences as training examples for a supervised classifier (Hoffmann et al., 2010; Wu and Weld, 2010; Hoffmann et al., 2011; Wang et al., 2011; Yao et al., 2011). A usual problem is that two related entities may co-occur in one sentence for many unrelated reasons. For example, *Barack Obama* is the president of the *United States*, but not every sentence including the two entities supports and states this relation. Much of the previous work uses heuristics, e.g. extracting sentences only from encyclopedic entries (Mintz et al.,

*Work done during an internship at Google Zurich.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 54–59,
Jeju, Republic of Korea, 8-14 July 2012.

2009; Hoffmann et al., 2011; Wang et al., 2011), or syntactic restrictions on the sentences and the entity mentions (Wu and Weld, 2010). These are usually defined manually and may need to be adapted to different languages and domains. Manually selected seeds can also be used (Ravichandran and Hovy, 2002; Kozareva and Hovy, 2010).

The main contribution of this work is presenting a variant of *distance supervision* for relation extraction where we do not use heuristics in the selection of the training data. Instead, we use topic models to discriminate between the patterns that are expressing the relation and those that are ambiguous and can be applied across relations. In this way, high-precision extraction patterns can be learned without the need of any manual intervention.

2 Unsupervised relational pattern learning

Similar to other distant supervision methods, our approach takes as input an existing knowledge base containing entities and relations, and a textual corpus. In this work it is not necessary for the corpus to be related to the knowledge base. In what follows we assume that all the relations studied are binary and hold between exactly two entities in the knowledge base. We also assume a dependency parser is available, and that the entities have been automatically disambiguated using the knowledge base as sense inventory.

One of the most important problems to solve in distant supervision approaches is to be able to distinguish which of the textual examples that include two related entities, e_i and e_j, are supporting the relation. This section describes a fully unsupervised solution to this problem, computing the probability that a pattern supports a given relation, which will allow us to determine the most likely relation expressed in any sentence. Specifically, if a sentence contains two entities, e_i and e_j, connected through a pattern w, our model computes the probability that the pattern is expressing any relation –$P(r|w)$– for any relation r defined in the knowledge base. Note that we refer to patterns with the symbol w, as they are the words in our topic models.

Preprocessing As a first step, the textual corpus is processed and the data is transformed in the following way: (a) the input corpus is parsed and entities are disambiguated; (b) for each relation r in the knowledge base, a new (initially empty) document collection C_r is created; (c) for each entity pair (e_i, e_j) which are related in the knowledge base, a new (initially empty) document D_{ij} is created; (d) for each sentence in the input corpus containing one mention of e_i and one mention of e_j, a new term is added to D_{ij} consisting of the context in which the two entities were seen in the document. This context may be a complex structure, such as the dependency path joining the two entities, but it is considered for our purposes as a single term; (e) for each relation r relating e_i with e_j, document D_{ij} is added to collection C_r. Note that if the two entities are related in different ways at the same time, an identical copy of the document D_{ij} will be added to the collection for all those relations.

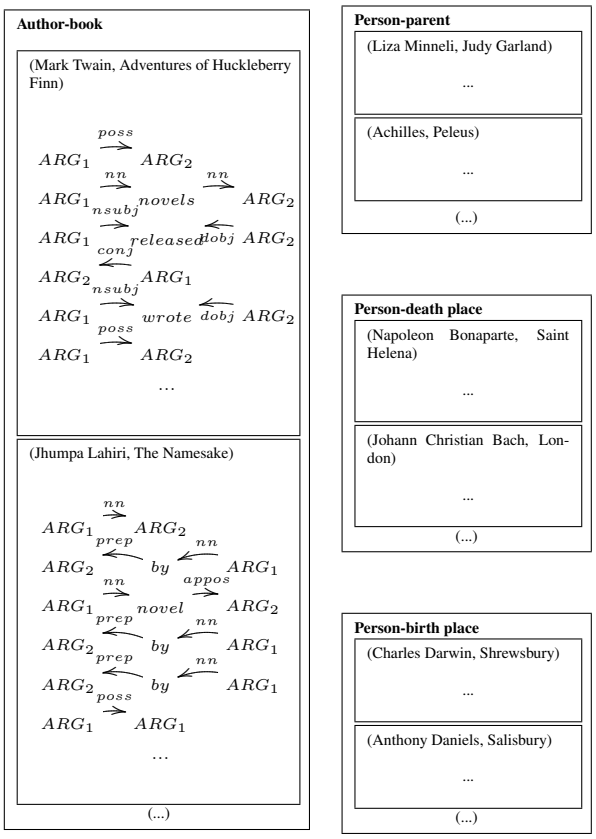


Figure 1: Example of a generated set of document collections from a news corpus for relation extraction. Larger boxes are document collections (relations), and inner boxes are documents (entity pairs). Document contain dependency patterns, which are *words* in the topic model.

Figure 1 shows a set of document collections gen-

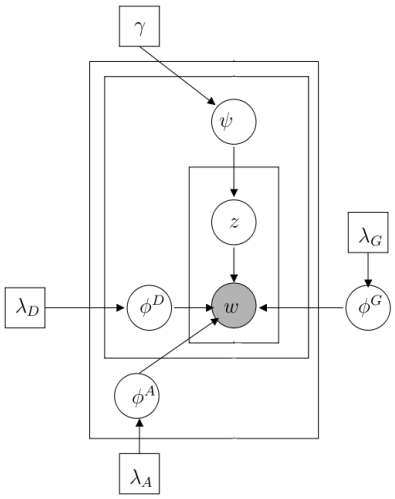


Figure 2: Plate diagram of the generative model used.

erated for three relations using this procedure. Each relation r has associated a different document collection, which contains one document associated to each entity pair from the knowledge base which is in relation r. The *words* in each document can be, for example, all the dependency paths that have been observed in the input textual corpus between the two related entities. Each document will contain some very generic paths (e.g. the two entities consecutive in the text) and some more specific paths.

Generative model Once these collections are built, we use the generative model from Figure 2 to learn the probability that a dependency path is conveying some relation between the entities it connects. This model is very similar to the one used by Haghighi and Vanderwende (2009) in the context of text summarization. w (the observed variable) represents a pattern between two entities. The topic model ϕ^G captures general patterns that appear for all relations. ϕ^D captures patterns that are specific about a certain entity pair, but which are not generalizable across all pairs with the same relation. Finally ϕ^A contains the patterns that are observed across most pairs related with the same relation. ϕ^A is the topic model of interest for us.

We use Gibbs sampling to estimate the different models from the source data. The topic assignments (for each pattern) that are the output of this process are used to estimate $P(r|w)$: when we observe pattern w, the probability that it conveys relation r.

3 Experiments and results

Settings We use Freebase as our knowledge base. It can be freely downloaded[1]. text corpus used contains 33 million English news articles that we downloaded between January 2004 and December 2011. A random sample of 3M of them is used for building the document collections on which to train the topic models, and the remaining 30M is used for testing. The corpus is preprocessed by identifying Freebase entity mentions, using an approach similar to (Milne and Witten, 2008), and parsing it with an inductive dependency parser (Nivre, 2006).

From the three million training documents, a set of document collections (one per relation) has been generated, by considering the sentences that contain two entities which are related in FreeBase through any binary relation and restricting to high-frequency 200 relations. Two ways of extracting patterns have been used: (a) **Syntactic**, taking the dependency path between the two entities, and (b) **Intertext**, taking the text between the two. In both cases, a topic model has been trained to learn the probability of a relation given a pattern w: $p(r|w)$. For λ we use symmetric Dirichlet priors $\lambda_G = 0.1$ and $\lambda_D = \lambda_{\mathcal{A}} = 0.001$, following the intuition that for the background the probability mass across patterns should be more evenly distributed. γ is set as (15, 15, 1), indicating in the prior that we expect more patterns to belong to the background and entity-pair-specific distributions due to the very noisy nature of the input data. These values have not been tuned.

As a baseline, using the same training corpus, we have calculated $p(r|w)$ using the maximum likelihood estimate: the number of times that a pattern w has been seen connecting two entities for which r holds divided by the total frequency of the pattern.

Extractions evaluation The patterns have been applied to the 30 million documents left for testing. For each pair of entities disambiguated as FreeBase entities, if they are connected through a known pattern, they are assigned $\arg\max_r p(r|w)$. We have randomly sampled 4,000 such extractions and sent them to raters. An extraction is to be judged correct if both it is correct in real life and the sentence from which it was extracted really supports it. We

[1] http://wiki.freebase.com/wiki/Data_dumps

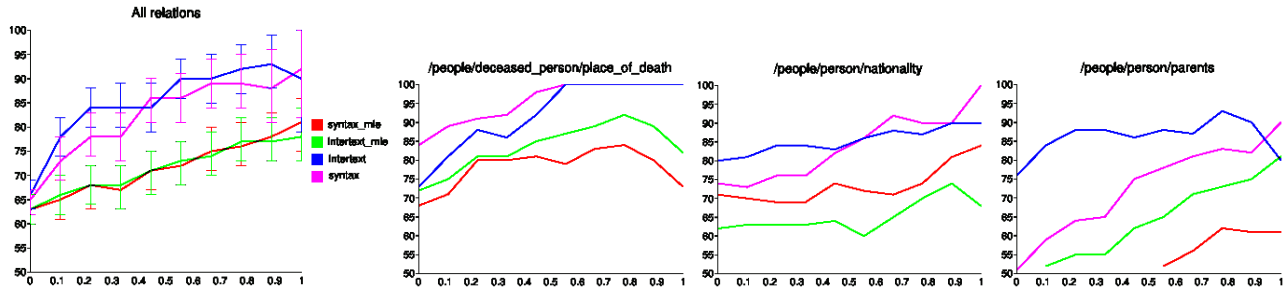


Figure 3: Evaluation of the extractions. X-axis has the threshold for $p(r|w)$, and Y-axis has the precision of the extractions as a percentage.

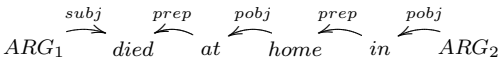
have collected three ratings per example and taken the majority decision. There was disagreement for 9.4% of the items on whether the sentence supports the relation, and for 20% of the items on whether the relation holds in the real world.

The results for different thresholds of $p(r|w)$ are shown in Figure 3. As can be seen, the MLE baselines (in red with syntactic patterns and green with intertext) perform consistently worse than the models learned using the topic models (in pink and blue). The difference in precision, aggregated across all relations, is statistically significant at 95% confidence for most of the thresholds.

Extractions aggregation We can take advantage of redundancy on the web to calculate a support metric for the extractions. In this experiment, for every extracted relation (r, e_1, e_2), for every occurrence of a pattern w_i connecting e_1 and e_2, we add up $p(r|w_i)$. Extractions that are obtained many times and from high-precision patterns will rank higher.

Table 1 describes the results of this aggregation. We have considered the top four highest-frequency relations for people. After aggregating all the extracted relations and ranking them by support, we have divided the evaluation set into two parts: (a) for relations that were not already in FreeBase, we evaluate the precision; (b) for extractions that were already in FreeBase, we take the top-confidence sentence identified and evaluate whether the sentence is providing support to the relation. For each of these, both syntactic patterns and intermediate-text patterns have been evaluated.

The results are very interesting: using syntax, *Death place* appears easy to extract new relations and to find support. The patterns obtained are quite unambiguous, e.g.



Relation	Unknown relations		Known relations	
	Correct relation P@50		Sentence support P@50	
	Syntax	*Intertext*	*Syntax*	*Intertext*
Parent	0.58	0.38	1.00	1.00
Death place	0.90	0.68	0.98	0.94
Birth place	0.38	0.56	0.54	0.98
Nationality	0.86	0.78	0.34	0.40

Table 1: Evaluation on aggregated extractions.

On the other hand, *birth place* and *nationality* have very different results for new relation acquisition vs. finding sentence support for new relations. The reason is that these relations are very correlated to other relations that we did not have in our training set. In the case of *birth place*, many relations refer to having an official position in the city, such as *mayor*; and for *nationality*, many of the patterns extract *presidents* or *ministers*. Not having *mayor* or *president* in our initial collection (see Figure 1), the support for these patterns is incorrectly learned. In the case of nationality, however, even though the extracted sentences do not support the relation (P@50 = 0.34 for intertext), the new relations extracted are mostly correct (P@50 = 0.86) as most presidents and ministers in the real world have the nationality of the country where they govern.

4 Conclusions

We have described a new distant supervision model with which to learn patterns for relation extraction with no manual intervention. Results are promising, we could obtain new relations that are not in FreeBase with a high precision for some relation types. It is also useful to extract support sentences for known relations. More work is needed in understanding which relations are compatible or overlapping and which ones can partially imply each other (such as *president-country* or *born_in-mayor*).

References

ACE. 2004. The automatic content extraction projects. http://projects.ldc.upenn.edu/ace.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI'07*.

D.T. Bollegala, Y. Matsuo, and M. Ishizuka. 2010. Relational duality: Unsupervised extraction of semantic relations between entities on the web. In *Proceedings of the 19th international conference on World wide web*, pages 151–160. ACM.

R. Bunescu and M. Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*, volume 6, pages 9–16.

S. Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, volume 2007, pages 708–716.

G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. 2004. The automatic content extraction (ace) program–tasks, data, and evaluation. In *Proceedings of LREC*, volume 4, pages 837–840. Citeseer.

A. Fader, S. Soderland, and O. Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of Empirical Methods in Natural Language Processing*.

A. Haghighi and L. Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics.

R. Hoffmann, C. Zhang, and D.S. Weld. 2010. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 286–295. Association for Computational Linguistics.

R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D.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-Volume 1*, pages 541–550. Association for Computational Linguistics.

G. Kasneci, M. Ramanath, F. Suchanek, and G. Weikum. 2009. The yago-naga approach to knowledge discovery. *ACM SIGMOD Record*, 37(4):41–47.

Z. Kozareva and E. Hovy. 2010. Learning arguments and supertypes of semantic relations using recursive patterns. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1482–1491. Association for Computational Linguistics.

C. Kwok, O. Etzioni, and D.S. Weld. 2001. Scaling question answering to the web. *ACM Transactions on Information Systems (TOIS)*, 19(3):242–262.

D. Li, S. Somasundaran, and A. Chakraborty. 2011. A combination of topic models with max-margin learning for relation detection.

P. McNamee and H.T. Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Text Analysis Conference (TAC)*.

D. Milne and I.H. Witten. 2008. Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.

M. Mintz, S. Bills, R. Snow, and D. 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: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

J. Nivre. 2006. Inductive dependency parsing. In *Text, Speech and Language Technology*, volume 34. Springer Verlag.

M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain. 2006. Organizing and searching the world wide web of facts-step one: the one-million fact extraction challenge. In *Proceedings of the National Conference on Artificial Intelligence*, page 1400. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

F. Pereira, A. Rajaraman, S. Sarawagi, W. Tunstall-Pedoe, G. Weikum, and A. Halevy. 2009. Answering web questions using structured data: dream or reality? *Proceedings of the VLDB Endowment*, 2(2):1646–1646.

H. Poon and P. Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 1–10. Association for Computational Linguistics.

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 41–47. Association for Computational Linguistics.

S. Sekine. 2006. On-demand information extraction. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 731–738. Association for Computational Linguistics.

Beth M. Sundheim and Nancy A. Chinchor. 1993. Survey of the message understanding conferences. In *HLT'93*.

I. Titov and A. Klementiev. 2011. A bayesian model for unsupervised semantic parsing. In *The 49th Annual Meeting of the Association for Computational Linguistics*.

C. Wang, J. Fan, A. Kalyanpur, and D. Gondek. 2011. Relation extraction with relation topics. In *Proceedings of Empirical Methods in Natural Language Processing*.

Daniel S. Weld, Fei Wu, Eytan Adar, Saleema Amershi, James Fogarty, Raphael Hoffmann, Kayur Patel, and Michael Skinner. 2008. Intelligence in wikipedia. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 1609–1614. AAAI Press.

F. Wu and D.S. Weld. 2010. Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127. Association for Computational Linguistics.

L. Yao, A. Haghighi, S. Riedel, and A. McCallum. 2011. Structured relation discovery using generative models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

A. Yates, M. Cafarella, M. Banko, O. Etzioni, M. Broadhead, and S. Soderland. 2007. Textrunner: Open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics.

Self-Disclosure and Relationship Strength in Twitter Conversations

JinYeong Bak, Suin Kim, Alice Oh
Department of Computer Science
Korea Advanced Institute of Science and Technology
Daejeon, South Korea
{jy.bak, suin.kim}@kaist.ac.kr, alice.oh@kaist.edu

Abstract

In social psychology, it is generally accepted that one discloses more of his/her personal information to someone in a strong relationship. We present a computational framework for automatically analyzing such self-disclosure behavior in Twitter conversations. Our framework uses text mining techniques to discover topics, emotions, sentiments, lexical patterns, as well as personally identifiable information (PII) and personally embarrassing information (PEI). Our preliminary results illustrate that in relationships with high relationship strength, Twitter users show significantly more frequent behaviors of self-disclosure.

1 Introduction

We often *self-disclose*, that is, share our emotions, personal information, and secrets, with our friends, family, coworkers, and even strangers. Social psychologists say that the degree of self-disclosure in a relationship depends on the strength of the relationship, and strategic self-disclosure can strengthen the relationship (Duck, 2007). In this paper, we study whether relationship strength has the same effect on self-disclosure of Twitter users.

To do this, we first present a method for computational analysis of self-disclosure in online conversations and show promising results. To accommodate the largely unannotated nature of online conversation data, we take a topic-model based approach (Blei et al., 2003) for discovering latent patterns that reveal self-disclosure. A similar approach was able to discover sentiments (Jo and Oh, 2011) and emotions (Kim et al., 2012) from user contents. Prior work on self-disclosure for online social networks has been from communications research (Jiang et al., 2011; Humphreys et al., 2010) which relies on human judgements for analyzing self-disclosure. The limitation of such research is that the data is small, so our approach of automatic analysis of self-disclosure will be able to show robust results over a much larger data set.

Analyzing relationship strength in online social networks has been done for Facebook and Twitter in (Gilbert and Karahalios, 2009; Gilbert, 2012) and for enterprise SNS (Wu et al., 2010). In this paper, we estimate relationship strength simply based on the duration and frequency of interaction. We then look at the correlation between self-disclosure and relationship strength and present the preliminary results that show a positive and significant correlation.

2 Data and Methodology

Twitter is widely used for conversations (Ritter et al., 2010), and prior work has looked at Twitter for different aspects of conversations (Boyd et al., 2010; Danescu-Niculescu-Mizil et al., 2011; Ritter et al., 2011). Ours is the first paper to analyze the degree of self-disclosure in conversational tweets. In this section, we describe the details of our Twitter conversation data and our methodology for analyzing relationship strength and self-disclosure.

2.1 Twitter Conversation Data

A Twitter conversation is a *chain* of tweets where two users are consecutively replying to each other's tweets using the Twitter *reply* button. We identified dyads of English-tweeting users who had at least

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 60–64,
Jeju, Republic of Korea, 8-14 July 2012.

three conversations from October, 2011 to December, 2011 and collected their tweets for that duration. To protect users' privacy, we anonymized the data to remove all identifying information. This dataset consists of 131,633 users, 2,283,821 chains and 11,196,397 tweets.

2.2 Relationship Strength

Research in social psychology shows that relationship strength is characterized by interaction frequency and closeness of a relationship between two people (Granovetter, 1973; Levin and Cross, 2004). Hence, we suggest measuring the relationship strength of the conversational dyads via the following two metrics. **Chain frequency** (CF) measures the number of conversational chains between the dyad averaged per month. **Chain length** (CL) measures the length of conversational chains between the dyad averaged per month. Intuitively, high CF or CL for a dyad means the relationship is strong.

2.3 Self-Disclosure

Social psychology literature asserts that self-disclosure consists of personal information and open communication composed of the following five elements (Montgomery, 1982).

Negative openness is how much disagreement or negative feeling one expresses about a situation or the communicative partner. In Twitter conversations, we analyze sentiment using the aspect and sentiment unification model (ASUM) (Jo and Oh, 2011), based on LDA (Blei et al., 2003). ASUM uses a set of seed words for an unsupervised discovery of sentiments. We use positive and negative emoticons from Wikipedia.org[1]. **Nonverbal openness** includes facial expressions, vocal tone, bodily postures or movements. Since tweets do not show these, we look at emoticons, 'lol' (laughing out loud) and 'xxx' (kisses) for these nonverbal elements. According to Derks et al. (2007), emoticons are used as substitutes for facial expressions or vocal tones in socio-emotional contexts. We also consider profanity as nonverbal openness. The methodology used for identifying profanity is described in the next section. **Emotional openness** is how much one discloses his/her feelings and moods. To measure this, we look for tweets that contain words that are identified as the most common expressions of feelings in blogs as found in Harris and Kamvar (2009). **Receptive openness** and **General-style openness** are difficult to get from tweets, and they are not defined precisely in the literature, so we do not consider these here.

2.4 PII, PEI, and Profanity

PII and PEI are also important elements of self-disclosure. Automatically identifying these is quite difficult, but there are certain topics that are indicative of PII and PEI, such as *family*, *money*, *sickness* and *location*, so we can use a widely-used topic model, LDA (Blei et al., 2003) to discover topics and annotate them using MTurk[2] for PII and PEI, and profanity. We asked the Turkers to read the conversation chains representing the topics discovered by LDA and have them mark the conversations that contain PII and PEI. From this annotation, we identified five topics for profanity, ten topics for PII, and eight topics for PEI. Fleiss kappa of MTurk result is 0.07 for PEI, and 0.10 for PII, and those numbers signify *slight agreement* (Landis and Koch, 1977). Table 1 shows some of the PII and PEI topics. The profanity words identified this way include *nigga, lmao, shit, fuck, lmfao, ass, bitch*.

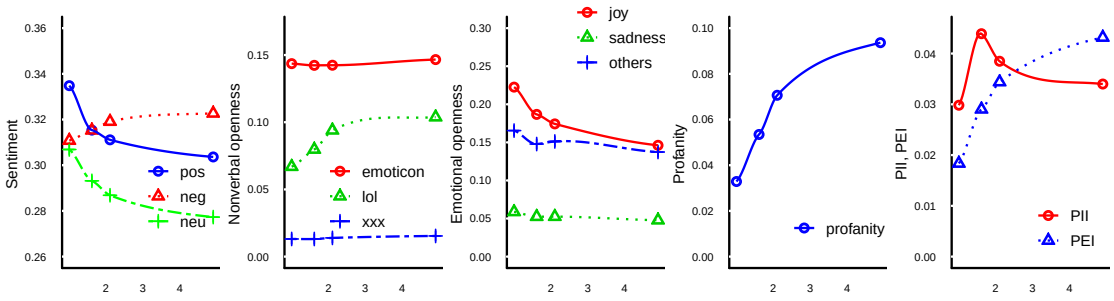
PII 1	PII 2	PEI 1	PEI 2	PEI 3
san	tonight	pants	teeth	family
live	time	wear	doctor	brother
state	tomorrow	boobs	dr	sister
texas	good	naked	dentist	uncle
south	ill	wearing	tooth	cousin

Table 1: PII and PEI topics represented by the high-ranked words in each topic.

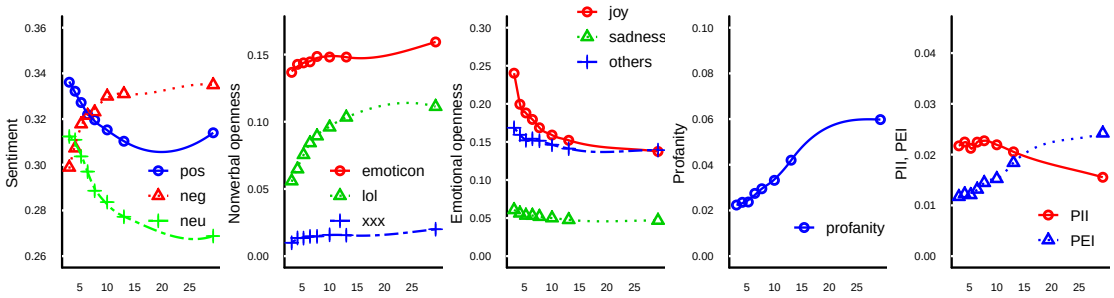
To verify the topic-model based approach to discovering PII and PEI, we tried supervised classification using SVM on document-topic proportions. Precision and recall are 0.23 and 0.21 for PII, and 0.30 and 0.23 for PEI. These results are not quite good, but this is a difficult task even for humans, and we had a low agreement among the Turkers. So our current work is in improving this.

[1] http://en.wikipedia.org/wiki/List of emoticons

[2] https://www.mturk.com



(a) Chain Frequency



(b) Conversation Length

Figure 1: Degree of self-disclosure depending on various relationship strength metrics. The x axis shows relationship strength according to tweeting behavior (chain frequency and chain length), and the y axis shows proportion of self-disclosure in terms of negative openness, emotional openness, profanity, and PII and PEI.

3 Results and Discussions

Chain frequency (CF) and chain length (CL) reflect the dyad's tweeting behaviors. In figure 1, we can see that the two metrics show similar patterns of self-disclosure. When two users have stronger relationships, they show more negative openness, nonverbal openness, profanity, and PEI. These patterns are expected. However, weaker relationships tend to show more PII and emotions. A closer look at the data reveals that PII topics are related to cities where they live, time of day, and birthday. This shows that the weaker relationships, usually new acquaintances, use PII to introduce themselves or send trivial greetings for birthdays. Higher emotional openness in weaker relationships looks strange at first, but similar to PII, emotion in weak relationships is usually expressed as greetings, reactions to baby or pet photos, or other shallow expressions.

It is interesting to look at outliers, dyads with very strong and very weak relationship groups. Table 3 summarizes the self-disclosure behaviors of these outliers. There is a clear pattern that stronger relationships show more nonverbal openness, negative openness, profanity use, and PEI. In figure 1, emotional openness does not differ for the strong and weak relationship groups. We can see why this is when we look at the topics for the strong and weak groups. Table 2 shows the topics that are most prominent in the strong relationships, and they include daily greetings, plans, nonverbal emotions such as 'lol', 'omg', and profanity. In weak relationships, the prominent topics illustrate the prevalence of initial getting-to-know conversations in Twitter. They welcome and greet each other about kids and pets, and offer sympathies about feeling bad.

str1	str2	weak1	weak2	weak3
lmao	sleep	following	ill	love
lmfao	bed	thanks	sure	thanks
shit	night	followers	soon	cute
ass	tired	welcome	better	aww
smh	awake	follow	want	pretty

Table 2: Topics that are most prominent in strong ('str') and weak relationships.

One interesting way to use our analysis is in iden-

	strong	weak
# relation	5,640	226,116
CF	14.56	1.00
CL	97.74	3.00
Emotion	0.21	0.22
Emoticon	0.162	0.134
lol	0.105	0.060
xxx	0.021	0.006
Pos Sent	0.31	0.33
Neg Sent	0.32	0.29
Neut Sent	0.27	0.29
Profanity	0.0615	0.0085
PII	0.016	0.019
PEI	0.022	0.013

Table 3: Comparing the top 1% and the bottom 1% relationships as measured by the combination of CF and CL. From 'Emotion' to PEI, all values are average proportions of tweets containing each self-disclosure behavior. Strong relationships show more negative sentiment, profanity, and PEI, and weak relationships show more positive sentiment and PII. 'Emotion' is the sum of all emotion categories and shows little difference.

tifying a rare situation that deviates from the general pattern, such as a dyad linked weakly but shows high self-disclosure. We find several such examples, most of which are benign, but some do show signs of risk for one of the parties. In figure 2, we show an example of a conversation with a high degree of self-disclosure by a dyad who shares only one conversation in our dataset spanning two months.

4 Conclusion and Future Work

We looked at the relationship strength in Twitter conversational partners and how much they self-disclose to each other. We found that people disclose more to closer friends, confirming the social psychology studies, but people show more positive sentiment to weak relationships rather than strong relationships. This reflects the social norm toward first-time acquaintances on Twitter. Also, emotional openness does not change significantly with relationship strength. We think this may be due to the inherent difficulty in truly identifying the emotions on Twitter. Identifying emotion merely based on keywords captures mostly shallow emotions, and deeper emotional openness either does not occur much on

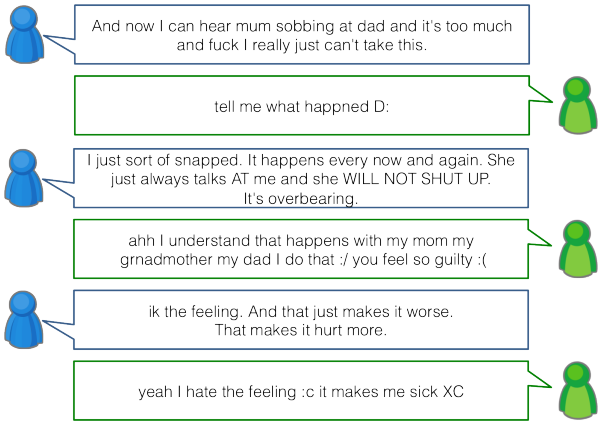


Figure 2: Example of Twitter conversation in a weak relationship that shows a high degree of self-disclosure.

Twitter or cannot be captures very well.

With our automatic analysis, we showed that when Twitter users have conversations, they control self-disclosure depending on the relationship strength. We showed the results of measuring the relationship strength of a Twitter conversational dyad with chain frequency and length. We also showed the results of automatically analyzing self-disclosure behaviors using topic modeling.

This is ongoing work, and we are looking to improve methods for analyzing relationship strength and self-disclosure, especially emotions, PII and PEI. For relationship strength, we will consider not only interaction frequency, but also network distance and relationship duration. For finding emotions, first we will adapt existing models (Vaassen and Daelemans, 2011; Tokuhisa et al., 2008) and suggest a new semi-supervised model. For finding PII and PEI, we will not only consider the topics, but also time, place and the structure of questions and answers. This paper is a starting point that has shown some promising research directions for an important problem.

5 Acknowledgment

We thank the anonymous reviewers for helpful comments. This research is supported by Korean Ministry of Knowledge Economy and Microsoft Research Asia (N02110403).

References

D.M. Blei, A.Y. Ng, and M.I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

D. Boyd, S. Golder, and G. Lotan. 2010. Tweet, tweet, retweet: Conversational aspects of retweeting on twitter. In *Proceedings of the 43rd Hawaii International Conference on System Sciences*.

C. Danescu-Niculescu-Mizil, M. Gamon, and S. Dumais. 2011. Mark my words!: linguistic style accommodation in social media. In *Proceedings of the 20th International World Wide Web Conference*.

D. Derks, A.E.R. Bos, and J. Grumbkow. 2007. Emoticons and social interaction on the internet: the importance of social context. *Computers in Human Behavior*, 23(1):842–849.

S. Duck. 2007. *Human Relationships*. Sage Publications Ltd.

E. Gilbert and K. Karahalios. 2009. Predicting tie strength with social media. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, pages 211–220.

E. Gilbert. 2012. Predicting tie strength in a new medium. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*.

M.S. Granovetter. 1973. The strength of weak ties. *American Journal of Sociology*, pages 1360–1380.

J. Harris and S. Kamvar. 2009. *We Feel Fine: An Almanac of Human Emotion*. Scribner Book Company.

L. Humphreys, P. Gill, and B. Krishnamurthy. 2010. How much is too much? privacy issues on twitter. In *Conference of International Communication Association, Singapore*.

L. Jiang, N.N. Bazarova, and J.T. Hancock. 2011. From perception to behavior: Disclosure reciprocity and the intensification of intimacy in computer-mediated communication. *Communication Research*.

Y. Jo and A.H. Oh. 2011. Aspect and sentiment unification model for online review analysis. In *Proceedings of International Conference on Web Search and Data Mining*.

S. Kim, J. Bak, and A. Oh. 2012. Do you feel what i feel? social aspects of emotions in twitter conversations. In *Proceedings of the AAAI International Conference on Weblogs and Social Media*.

J.R. Landis and G.G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174.

D.Z. Levin and R. Cross. 2004. The strength of weak ties you can trust: The mediating role of trust in effective knowledge transfer. *Management science*, pages 1477–1490.

B.M. Montgomery. 1982. Verbal immediacy as a behavioral indicator of open communication content. *Communication Quarterly*, 30(1):28–34.

A. Ritter, C. Cherry, and B. Dolan. 2010. Unsupervised modeling of twitter conversations. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 172–180.

A. Ritter, C. Cherry, and W.B. Dolan. 2011. Data-driven response generation in social media. In *Proceedings of EMNLP*.

R. Tokuhisa, K. Inui, and Y. Matsumoto. 2008. Emotion classification using massive examples extracted from the web. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 881–888.

F. Vaassen and W. Daelemans. 2011. Automatic emotion classification for interpersonal communication. *ACL HLT 2011*, page 104.

A. Wu, J.M. DiMicco, and D.R. Millen. 2010. Detecting professional versus personal closeness using an enterprise social network site. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems*.

Genre Independent Subgroup Detection in Online Discussion Threads: A Pilot Study of Implicit Attitude using Latent Textual Semantics

Pradeep Dasigi pd2359@columbia.edu
Weiwei Guo weiwei@cs.columbia.edu
Mona Diab mdiab@ccls.columbia.edu
Center for Computational Learning Systems, Columbia University

Abstract

We describe an unsupervised approach to the problem of automatically detecting subgroups of people holding similar opinions in a discussion thread. An intuitive way of identifying this is to detect the attitudes of discussants towards each other or named entities or topics mentioned in the discussion. Sentiment tags play an important role in this detection, but we also note another dimension to the detection of people's attitudes in a discussion: if two persons share the same opinion, they tend to use similar language content. We consider the latter to be an implicit attitude. In this paper, we investigate the impact of implicit and explicit attitude in two genres of social media discussion data, more formal wikipedia discussions and a debate discussion forum that is much more informal. Experimental results strongly suggest that implicit attitude is an important complement for explicit attitudes (expressed via sentiment) and it can improve the sub-group detection performance independent of genre.

1 Introduction

There has been a significant increase in discussion forum data in online media recently. Most of such discussion threads have a clear debate component in them with varying levels of formality. Automatically identifying the groups of discussants with similar attitudes, or subgroup detection, is an interesting problem which allows for a better understanding of the data in this genre in a manner that could directly benefit Opinion Mining research as well as Community Mining from Social Networks.

A straight-forward approach to this problem is to apply Opinion Mining techniques, and extract each discussant's attitudes towards other discussants and entities being discussed. But the challenge is that Opinion Mining is not mature enough to extract all the correct opinions of discussants. In addition, without domain knowledge, using unsupervised techniques to do this is quite challenging.

On observing interactions from these threads, we believe that there is another dimension of attitude which is expressed implicitly. We find that people sharing the same opinion tend to speak about the same topics even though they do not explicitly express their sentiment. We refer to this as *Implicit Attitude*. One such example may be seen in the two posts in Table 1. It can be seen that even though discussants *A* and *B* do not express explicit sentiments, they hold similar views. Hence it can be said that there is an agreement in their implicit attitudes.

Attempting to find a surface level word similarity between posts of two discussants is not sufficient as there are typically few overlapping words shared among the posts. This is quite significant a problem especially given the relative short context of posts. Accordingly, in this work, we attempt to model the implicit latent similarity between posts as a means of identifying the implicit attitudes among discussants. We apply variants on Latent Dirichelet Allocation (LDA) based topic models to the problem (Blei et al., 2003).

Our goal is identify subgroups with respect to discussants' attitudes towards each other, the entities and topics in a discussion forum. To our knowledge, this is the first attempt at using text similarity as an indication of user attitudes. We investigate the influence of the explicit and implicit attitudes on two genres of data, one more formal than the other. We find an interesting trend. Explicit attitude alone

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 65–69,
Jeju, Republic of Korea, 8-14 July 2012.

as a feature is more useful than implicit attitude in identifying sub-groups in informal data. But in the case of formal data, implicit attitude yields better results. This may be due to the fact that in informal data, strong subjective opinions about entities/events or towards other discussants are expressed more explicitly. This is generally not the case in the formal genre where ideas do not have as much sentiment associated with them, and hence the opinions are more "implicit". Finally, we observe that combining both kinds of features improves performance of our systems for both genres.

2 Related Work

Substantial research exists in the fields of Opinion Identification and Community Mining that is related to our current work. (Ganapathibhotla and Liu, 2008) deal with the problem of finding opinions from comparative sentences. Many previous research efforts related to Opinion Target Identification (Hu and Liu, 2004; Kobayashi et al., 2007; Jakob and Gurevych, 2010), focus on the domain of product reviews where they exploit the genre in multiple ways. Somasundaran and Wiebe (2009) used unsupervised methods to identify stances in online debates. They mine the web to find associations indicative of opinions and combine them with discourse information. Their problem essentially deals with the debate genre and finding the stance of an individual given two options. Ours is a more general problem since we deal with discussion data in general and not debates on specific topics. Hence our aim is to identify multiple groups, not just two.

In terms of Sentiment Analysis, the work done by Hassan et al.(2010) in using part-of-speech and dependency structures to identify polarities of attitudes is similar to our work. But they predict binary polarities in attitudes, and our goal of identification of sub-groups is a more general problem in that we aim at identifying multiple subgroups.

3 Approach

We tackle the problem using Vector Space Modeling techniques to represent the discussion threads. Each vector represents a discussant in the thread creating an *Attitude Profile* (AP). We use a clustering algorithm to partition the vector space of APs into multiple sub-groups. The idea is that resulting clusters would comprise sub-groups of discussants with similar attitudes.

3.1 Basic Features

We use two basic features, namely Negative and Positive sentiment towards specific discussants and entities like in the work done by (Abu-Jbara et al., 2012). We start off by determining sentences that express attitude in the thread, attitude sentences (AS). We use OpinionFinder (Wilson et al., 2005) which employs negative and positive polarity cues. For determining discussant sentiment, we need to first identify who the target of their sentiment is: another discussant, or an entity, where an entity could be a topic or a person not participating in the discussion. **Sentiment toward another discussant:** This is quite challenging since explicit sentiment expressed in a post is not necessarily directed towards another discussant to whom it is a reply. It is possible that a discussant may be replying to another poster but expressing an attitude towards a third entity or discussant. However as a simplifying assumption, similar to the work of (Hassan et al., 2010), we adopt the view that replies in the sentences that are determined to be attitudinal and contain second-person pronouns (you, your, yourself) are assumed to be directed towards the recipients of the replies. **Sentiment toward an entity:** We again adopt a simplifying view by modeling all the named entities in a sentence without heeding the roles these entities play, i.e. whether they are targets or not. Accordingly, we extract all the named entities in a sentence using Stanford's Name Entity Recognizer (Finkel et al., 2005). We only focus on Person and Organization named entities.

3.2 Extracting Implicit Attitudes

We define implicit attitudes as the semantic similarity between texts comprising discussant utterances or posts in a thread. We cannot find enough overlapping words between posts, since some posts are very short. Hence we apply LDA (Blei et al., 2003) on texts to extract latent semantics of texts. We split text into sentences, i.e., each sentence is treated as a single document. Accordingly, each sentence is represented as a K-dimension vector. By computing the similarity on these vectors, we obtain a more accurate semantic similarity.

A: *There are a few other directors in the history of cinema who have achieved such a singular and consistent worldview as Kubrick. His films are very philosophically deep, they say something about everything, war, crime, relationships, humanity, etc.*

B: *All of his films show the true human nature of man and their inner fights and all of them are very philosophical. Alfred was good in suspense and all, but his work is not as deep as Kubrick's*

Table 1: Example of Agreement based on Implicit Attitude

	WIKI	CD
Median No. of Discussants (n)	6	29
Predicted No. of Clusters ($\lceil\sqrt{\frac{n}{2}}\rceil$)	2	4
Median No. of Actual Classes	3	3

Table 2: Number of Clusters

3.3 Clustering Attitude Space

A tree-based (hierarchical) clustering algorithm, SLINK (Sibson, 1973) is used to cluster the vector space. Cosine Similarity between the vectors is used as the inter-data point similarity measure for clustering.[1] We choose the number of clusters to be $\lceil\sqrt{\frac{n}{2}}\rceil$, described as the rule of thumb by (Mardia et al., 1979), where n is the number of discussants in the group. This rule seems to be validated by the fact that in the data sets with which we experiment, we note that the predicted number of clusters according to this rule and the classes identified in the gold data are very close as illustrated in Table 2. On average we note that the gold data has the number of classes per thread to be roughly 2-5.

4 Data

We use data from two online forums - Create Debate [CD][2] and discussions from Wikipedia [WIKI][3]. There is a significant difference in the kind of discussions in these two sources. Our WIKI data comprises 117 threads crawled from Wikipedia. It is relatively formal with short threads. It does not have much negative polarity and discussants essentially discuss the Wikipedia page in question. Hence it is closer to an academic discussion forum. The threads are manually annotated with sub-group information. Given a thread, the annotator is asked to identify if there are any sub-groups among the discussants with similar opinions, and if yes, the membership of those subgroups.

Property	WIKI	CD
Threads	117	34
Posts per Thread	15.5	112
Sentences per Post	4.5	7.7
Tokens per Post	78.9	118.3
Word Types per Post	11.1	10.6
Discussants per Thread	6.5	34.15
Entities Discovered per Thread	6.15	32.7

Table 3: Data Statistics

On the other hand, CD is a forum where people debate a specific topic. The CD data we use comprises 34 threads. It is more informal (with pervasive negative language and personal insults) than WIKI and has longer threads. It is closer to the debate genre. It has a poll associated with every debate. The votes cast by the discussants in the poll are used as the class labels for our experiments. Detailed statistics related to both the data sets and a comparison can be found in Table 3.

5 Experimental Conditions

The following three features represent discussant attitudes:

• Sentiment towards other discussants (SD) - This corresponds to $2 * n$ dimensions in the *Attitude Profile* (AP) vector, n being the number of discussants in the thread. This is because there are two polarities and n possible targets. The value representing this feature is the number of sentences with the respective polarity – negative or positive – towards the particular discussant.

• Sentiment towards entities in discussion (SE) - Number of dimensions corresponding to this feature is $2 * e$, where e is the number of entities discovered. Similar to SD, the value taken by this feature is the number of sentences in which that specific polarity is shown by the discussant towards the entity.

• Implicit Attitude (IA) - $n * t$ dimensions are expressed using this feature, where t is the number of topics that the topic model contains. This means that the AP of every discussant contains the topic model distribution of his/her interactions with every other member in the thread. Hence, the topics in the interation between the given discussant and other members in the thread are being modeled here. Accord-

[1] We also experimented with K-means (MacQueen, 1967) and found that it yields worse results compared to SLINK. There is a fundamental difference between the two algorithms. Where as K-Means does a random initialization of clusters, SLINK is a deterministic algorithm. The difference in the performance may be attributed to the fact that the number of initial data points is too small for random initialization. Hence, tree based clustering algorithms are more well suited for the current task.

[2] http://www.createdebate.com

[3] en.wikipedia.org

ingly, high vector similarity due to IA between two members in a thread means that they discussed similar topics with the same people in the thread. In our experiments, we set $t = 50$. We use the Gibbs sampling based LDA (Griffiths and Steyvers, 2004). The LDA model is built on definitions of two online dictionaries WordNet, and Wiktionary, in addition to the Brown corpus (BC). To create more context, each sentence from BC is treated as a document. The whole corpus contains 393,667 documents and 5,080,369 words.

The degree of agreement among discussants in terms of these three features is used to identify subgroups among them. Our experiments are aimed at investigating the effect of explicit attitude features (SD and SE) in comparison with implicit feature (IA) and how they perform when combined. So the experimental conditions are: the three features in isolation, each of the explicit features SD and SE together with IA, and then all three features together.

SWD-BASE: As a baseline, we employ a simple word frequency based model to capture topic distribution, Surface Word Distribution (SWD). SWD is still topic modeling in the vector space, but the dimensions of the vectors are the frequencies of all the unique words used by the discussant in question.

RAND-BASE: We also apply a very simple baseline using random assignment of discussants to groups, however the number of clusters is determined by the rule of thumb described in Section 3.3.

6 Results and Analysis

Three metrics are used for evaluation, as described in (Manning et al., 2008): Purity, Entropy and F-measure. Table 4 shows the results of the 9 experimental conditions. The following observations can be made: All the individual conditions SD, SE and IA clearly outperform SWD-BASE. All the experimental conditions outperform RAND-BASE which indicates that using clustering is contributing positively to the problem. SE performs worse than SD across both datasets CD and WIKI. This may be due to two reasons: Firstly, since the problem is of clustering the discussant space, SD should be a better indicator than SE. Secondly, as seen from the comparison in Table 5, there are more polarized sentences indicating SD than SE. IA clearly outperforms SD, SE and SD+SE in the case of WIKI. In the case of CD, it is exactly the opposite. This is an interesting result and we believe it is mainly due to the genre of the data. Explicit expression of sentiment usually increases with the increase in the informal nature of discussions. Hence IA is more useful in WIKI which is more formal compared to CD, where there is less overt sentiment expression. We note the same trend with the SWD-BASE where performance on WIKI is much better than its performance on CD. This also suggests that WIKI might be an easier data set. A qualitative comparison of the inter-discussant relations can be gleaned from Table 5. There is significantly more negative language than positive language in CD when compared with the ratios of negative to positive language in WIKI, which are almost the same. The best results overall are yielded from the combination of IA with SD and SE, the implicit and explicit features together for both data sets, which suggests that Implicit and explicit attitude features complement each other capturing more information than each of them individually.

Property	WIKI	CD
Positive Sentences towards Discussants	5.15	17.94
Negative Sentences towards Discussants	6.75	40.38
Positive Sentences towards Entities	1.65	8.85
Negative Sentences towards Entities	1.59	8.53

Table 5: Statistics of the Attitudinal Sentences per each Thread in the two data sets

7 Conclusions

We proposed the use of LDA based topic modeling as an implicit agreement feature for the task of identifying similar attitudes in online discussions. We specifically applied latent modeling to the problem of sub-group detection. We compared this with explicit sentiment features in different genres both in isolation and in combination. We highlighted the difference in genre in the datasets and the necessity for capturing different forms of information from them for the task at hand. The best yielding condition in both the dat sets combines implicit and explicit features suggesting that there is a complementarity between the two tpes of feaures.

Acknowledgement

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.

Condition	WIKI			CD		
	Purity	**Entropy**	**F-measure**	**Purity**	**Entropy**	**F-measure**
RAND-BASE	0.6745	0.5629	0.6523	0.3986	0.9664	0.407
SWD-BASE	0.7716	0.4746	0.6455	0.4514	0.9319	0.4322
SD	0.8342	0.3602	0.667	0.8243	0.3942	0.5964
SE	0.8265	0.3829	0.6554	0.7933	0.4216	0.5818
SD+SE	0.8346	0.3614	0.6649	0.82	0.3851	0.6039
IA	0.8527	0.3209	0.6993	0.787	0.3993	0.5891
SD+IA	0.8532	0.3199	0.6977	0.8487	0.3328	0.6152
SE+IA	0.8525	0.3216	0.7015	0.7884	0.3986	0.591
SD+SE+IA	**0.8572**	**0.3104**	**0.7032**	**0.8608**	**0.3149**	**0.6251**

Table 4: Experimental Results

References

Amjad Abu-Jbara, Pradeep Dasigi, Mona Diab, and Dragomir Radev. 2012. Subgroup detection in ideological discussions. In *Proceedings of the 5oth Annual Meeting of ACL*.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics*.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101.

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,*.

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*.

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*.

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*.

J. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*.

Christopher D. Manning, Prabhakar Raghavan, , and Hinrich Schtze. 2008. . *2008. Introduction to Information Retrieval*. Cambridge University Press, New York, NY,USA.

K. V. Mardia, J. T. Kent, and J. M. Bibby. 1979. *Multivariate Analysis*. Publisher.

R. Sibson. 1973. Slink: An optimally efficient algorithm for the single-link cluster method. In *The Computer Journal (1973) 16 (1): 30-34.*

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*.

Theresa Wilson, Paul Hoffmann, Swapna Somasundaran, Jason Kessler, JanyceWiebe, Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005. Opinionfinder: A system for subjectivity analysis. In *Proceedings of HLT/EMNLP 2005 Demonstration*.

Learning to Temporally Order Medical Events in Clinical Text

Preethi Raghavan*, **Eric Fosler-Lussier***, **and Albert M. Lai**†
*Department of Computer Science and Engineering
†Department of Biomedical Informatics
The Ohio State University, Columbus, Ohio, USA
{raghavap, fosler}@cse.ohio-state.edu, albert.lai@osumc.edu

Abstract

We investigate the problem of ordering medical events in unstructured clinical narratives by learning to rank them based on their time of occurrence. We represent each medical event as a time duration, with a corresponding start and stop, and learn to rank the starts/stops based on their proximity to the admission date. Such a representation allows us to learn all of Allen's temporal relations between medical events. Interestingly, we observe that this methodology performs better than a classification-based approach for this domain, but worse on the relationships found in the Timebank corpus. This finding has important implications for styles of data representation and resources used for temporal relation learning: clinical narratives may have different language attributes corresponding to temporal ordering relative to Timebank, implying that the field may need to look at a wider range of domains to fully understand the nature of temporal ordering.

1 Introduction

There has been considerable research on learning temporal relations between events in natural language. Most learning problems try to classify event pairs as related by one of Allen's temporal relations (Allen, 1981) i.e., *before, simultaneous, includes/during, overlaps, begins/starts, ends/finishes* and their inverses (Mani et al., 2006). The Timebank corpus, widely used for temporal relation learning, consists of newswire text annotated for events, temporal expressions, and temporal relations between events using TimeML (Pustejovsky et al., 2003). In Timebank, the notion of an "event" primarily consists of verbs or phrases that denote change in state. However, there may be a need to rethink how we learn temporal relations between events in different domains. Timebank, its features, and established learning techniques like classification, may not work optimally in many real-world problems where temporal relation learning is of great importance.

We study the problem of learning temporal relations between medical events in clinical text. The idea of a medical "event" in clinical text is very different from events in Timebank. Medical events are temporally-associated concepts in clinical text that describe a medical condition affecting the patient's health, or procedures performed on a patient. Learning to temporally order events in clinical text is fundamental to understanding patient narratives and key to applications such as longitudinal studies, question answering, document summarization and information retrieval with temporal constraints. We propose learning temporal relations between medical events found in clinical narratives by learning to rank them. This is achieved by representing medical events as time durations with starts and stops and ranking them based on their proximity to the admission date.[1] This implicitly allows us to learn all of Allen's temporal relations between medical events.

In this paper, we establish the need to rethink the methods and resources used in temporal relation learning, as we demonstrate that the resources widely used for learning temporal relations in newswire text do not work on clinical text. When we model the temporal ordering problem in clinical text as a ranking problem, we empirically show that it outperforms classification; we perform similar experiments with Timebank and observe the opposite conclusion (classification outperforms ranking).

[1]The admission date is the only explicit date always present in each clinical narrative.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 70–74,
Jeju, Republic of Korea, 8-14 July 2012.

e1 *before* e2 e1.start e1.stop e2.start e2.stop	e1 *equals* e2 e1.start; e2.start e1.stop; e2.stop
e1 *overlaps with* e2 e1.start e2.start e1.stop e2.stop	e1 *starts* e2 e1.start; e2.start e1.stop e2.stop
e2 *during* e1 e1.start e2.start e2.stop e1.stop	e2 *finishes* e1 e1.start e2.start e1.stop; e2.stop

Table 1: Allen's temporal relations between medical events can be realized by ordering the starts and stops

HISTORY PHYSICAL DATE: **09/01/2007**
NAME: Smith Daniel T MR#: XXX-XX-XXXX
ATTENDING PHYSICIAN: John Payne MD DOB: 03/10/1940
HISTORY OF PRESENT ILLNESS
The patient is a 67-year-old Caucasian male with a history of paresis secondary to back injury who is bedridden status post colostomy and PEG tube who was brought by EMS with a history of fever. The patient gives a history of fever on and off associated with chills for the last 1 month. He does give a history of decubitus ulcer on the back but his main complaint is fever associated with epigastric discomfort.
PAST MEDICAL HISTORY
Significant for polymicrobial infection in the blood as well as in the urine in July 2007 history of back injury with paraparesis. He is status post PEG tube and colostomy tube.
REVIEW OF SYSTEMS
Positive for decubitus ulcer. No cough. There is fever. No shortness of breath.
PHYSICAL EXAMINATION
On physical exam the patient is a debilitated malnourished gentleman in mild distress. Abdomen showed PEG tube with discharging pus and there are multiple scars one in the midline. It had a healing wound. Bowel sounds were present. Extremities revealed pain and atrophied muscles in the lower extremities with decubitus ulcer which had a transparent bandage in the decubitus area which was stage 2-3. CNS - The patient is alert and awake x3. There was good power in both upper extremities. Cranial nerves II-XII grossly intact.

Figure 1: Excerpt from a sanitized clinical narrative (history & physical report) with medical events underlined.

2 Related Work

The Timebank corpus provides hand-tagged features, including tense, aspect, modality, polarity and event class. There have been significant efforts in machine learning of temporal relations between events using these features and a wide range of other features extracted from the Timebank corpus (Mani et al., 2006; Chambers et al., 2007; Lapata and Lascarides, 2011). The SemEval/TempEval (Verhagen et al., 2009) challenges have often focused on temporal relation learning between different types of events from Timebank. Zhou and Hripcsak (2007) provide a comprehensive survey of temporal reasoning with clinical data. There has also been some work in generating annotated corpora of clinical text for temporal relation learning (Roberts et al., 2008; Savova et al., 2009). However, none of these corpora are freely available. Zhou et al. (2006) propose a Temporal Constraint Structure (TCS) for medical events in discharge summaries. They use rule-based methods to induce this structure.

We demonstrate the need to rethink resources, features and methods of learning temporal relations between events in different domains with the help of experiments in learning temporal relations in clinical text. Specifically, we observe that we get better results in learning to rank chains of medical events to derive temporal relations (and their inverses) than learning a classifier for the same task.

The problem of learning to rank from examples has gained significant interest in the machine learning community, with important similarities and differences with the problems of regression and classification (Joachims et al., 2007). The joint cumulative distribution of many variables arises in problems of learning to rank objects in information retrieval and various other domains. To the best of our understanding, there have been no previous attempts to learn temporal relations between events using a ranking approach.

3 Representation of Medical Events (MEs)

Clinical narratives contain unstructured text describing various MEs including conditions, diagnoses and tests in the history of a patient, along with some information on when they occurred. Much of the temporal information in clinical text is implicit and embedded in relative temporal relations between MEs. A sample excerpt from a note is shown in Figure 1. MEs are temporally related both qualitatively (e.g., *paresis before colostomy*) and quantitatively (e.g. *chills 1 month before admission*). Relative time may be more prevalent than absolute time (e.g., *last 1 month, post colostomy* rather than *on July 2007*). Temporal expressions may also be fuzzy where *history* may refer to an event *1 year ago* or *3 months ago*. The relationship between MEs and time is complicated. MEs could be recurring or continuous vs. discrete date or time, such as *fever* vs. *blood in urine*. Some are long lasting vs. short-lived, such as *cancer, leukemia* vs. *palpitations*.

We represent MEs of any type of in terms of their time duration. The idea of time duration based representation for MEs is in the same spirit as TCS (Zhou et al., 2006). We break every ME *me* into *me.start* and *me.stop*. Given the ranking of all starts and stops, we can now compose every one of Allen's temporal relations (Allen, 1981). If it is clear from context that only the start or stop of a ME can be determined, then only that is considered. For instance, *"history of paresis secondary to back injury who is bedridden status post colostomy"* indicates the start of *paresis* is in the past history of the patient prior

to *colostomy*. We only know about *paresis.start* relative to other MEs and may not be able determine *paresis.stop*. For recurring and continuous events like *chills* and *fever*, if the time period of recurrence is continuous (*last 1 month*), we consider it to be the time duration of the event. If not continuous, we consider separate instances of the ME. For MEs that are associated with a fixed date or time, the start and stop are assumed to be the same (e.g., *polymicrobial infection in the blood as well as in the urine* in July 2007). In case of negated events like *no cough*, we consider *cough* as the ME with a negative polarity. Its start and stop time are assumed to be the same. Polarity allows us to identify events that actually occurred in the patient's history.

4 Ranking Model and Experiments

Given a patient with multiple clinical narratives, our objective is to induce a partial temporal ordering of all medical events in each clinical narrative based on their proximity to a reference date (admission).

The training data consists of medical event (ME) chains, where each chain consists of an instance of the start or stop of a ME belonging to the same clinical narrative along with a rank. The assumption is that the MEs in the same narrative are more or less semantically related by virtue of narrative discourse structure and are hence considered part of the same ME chain. The rank assigned to an instance indicates the temporal order of the event instance in the chain. Multiple MEs could occupy the same rank. Based on the rank of the starts and stops of event instances relative to other event instances, the temporal relations between them can be derived as indicated in Table 1. Our corpus for ranking consisted of 47 clinical narratives obtained from the medical center and annotated with MEs, temporal expressions, relations and event chains. The annotation agreement across our team of annotators is high; all annotators agreed on 89.5% of the events and our overall inter-annotator Cohen's kappa statistic (Conger, 1980) for MEs was 0.865. Thus, we extracted 47 ME chains across 4 patients. The distribution of MEs across event chains and chains across patients (p) is as as follows. p1 had 5 chains with 68 MEs, p2 had 9 chains with 90 MEs, p3 had 20 chains with 119 MEs and p4 had 13 chains with 82 MEs. The distribution of chains across different types of clinical narratives is shown in Figure 2. We construct a vector of features, from the manually annotated corpus, for each medical event instance. Although

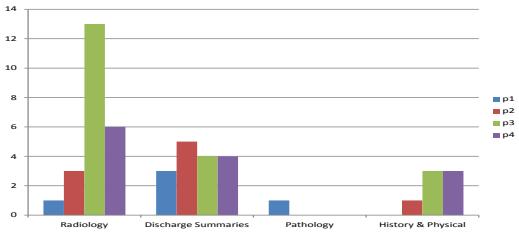


Figure 2: Distribution of the 47 medical event chains derived from discharge summaries, history and physical reports, pathology and radiology notes across the 4 patients.

there is no real query in our set up, the admission date for each chain can be thought of as the query "date" and the MEs are ordered based on how close or far they are from each other and the admission date. The features extracted for each ME include the the type of clinical narrative, section information, ME polarity, position of the medical concept in the narrative and verb pattern. We extract temporal expressions linked to the ME like *history, before admission, past, during examination, on discharge, after discharge, on admission.* Temporal references to specific times like *next day, previously* are resolved and included in the feature set. We also extract features from each temporal expression indicating its closeness to the admission date. Differences between each explicit date in the narrative is also extracted. The UMLS(Bodenreider, 2004) semantic category of each medical concept is also included based on the intuition that MEs of a certain semantic group may occur closer to admission. We tried using features like the tense of ME or the verb preceding the ME (if any), POS tag in ranking. We found no improvement in accuracy upon their inclusion.

In addition to the above features, we also anchor each ME to a coarse time-bin and use that as a feature in ranking. We define the following sequence of time-bins centered around admission, {*way before admission, before admission, on admission, after admission, after discharge*}. The time-bins are learned using a linear-chain CRF,[2] where the observation sequence is MEs in the order in which they appear in a clinical narrative, and the state sequence is the corresponding label sequence of time-bins.

We ran ranking experiments using SVM-rank (Joachims, 2006), and based on the ranking score assigned to each start/stop instance, we derive the relative temporal order of MEs in a chain.[3] This in turn allows us to infer temporal relations between

[2] http://mallet.cs.umass.edu/sequences.php

[3] In evaluating *simultaneous*, ± 0.05 difference in ranking score of starts/stops of MEs is counted as a match.

Relation	Clinical Text		Timebank	
	Ranking	Classifier	Ranking	Classifier
begins	81.21	73.34	52.63	58.82
ends	76.33	69.85	61.32	82.87
simulatenous	85.45	71.31	50.23	56.58
includes	83.67	74.20	59.56	60.65
before	88.3	77.14	61.34	70.38

Table 2: Per-class accuracy (%) for ranking, classification on clinical text and Timebank. We merge class ibefore into before.

all MEs in a chain. The ranking error on the test set is 28.2%. On introducing the time-bin feature, the ranking error drops to 16.8%. The overall accuracy of ranking MEs on including the time-bin feature is 82.16%. Each learned relation is now compared with the pairwise classification of temporal relations between MEs. We train a SVM classifier (Joachims, 1999) with an RBF kernel for pairwise classification of temporal relations. The average classification accuracy for clinical text using the same feature set is 71.33%. We used Timebank (v1.1) for evaluation, 186 newswire documents with 3345 event pairs. We traverse transitive relations between events in Timebank, increasing the number of event-event links to 6750 and create chains of related events to be ranked. Classification works better on Timebank, resulting in an overall accuracy of 63.88%, but ranking gives only 55.41% accuracy. All classification and ranking results from 10-fold cross validation are presented in Table 2.

5 Discussion

In ranking, the objective of learning is formalized as minimizing the fraction of swapped pairs over all rankings. This model is well suited to the features that are available in clinical text. The assumption that all MEs in a clinical narrative are temporally related allows us to totally order events within each narrative. This works because a clinical narrative usually has a single protagonist, the patient. This assumption, along with the availability of a fixed reference date in each narrative, allows us to effectively extract features that work in ranking MEs. However, this assumption does not hold in newswire text: there tend to be multiple protagonists, and it may be possible to totally order only events that are linked to the same protagonist. Ranking implicitly allows us to learn the transitive relations between MEs in the chain. Ranking ME starts/ stops captures relations like *includes* and *begins* much better than classification, primarily because of the date difference and time-bin difference features. However, the hand-tagged features available in Timebank are not suited for this kind of model. The features work well with classification but are not sufficiently informative to learn time durations using our proposed event representation in a ranking model. Features like "tense" that are used for temporal relation learning in Timebank are not very useful in ME ordering. Tense is a temporal linguistic quality expressing the time at, or during which a state or action denoted by a verb occurs. In most cases, MEs are not verbs (e.g., *colostomy*). Even if we consider verbs co-occurring with MEs, they are not always accurately reflective of the MEs' temporal nature. Moreover, in discharge summaries, almost all MEs or co-occurring verbs are in the past tense (before the discharge date). This is complicated by the fact that the reference time/ ME with respect to which the tense of the verb is expressed is not always clear. Based on the type of clinical narrative, when it was generated, the reference date for the tense of the verb could be in the patient's history, admission, discharge, or an intermediate date between admission and discharge. For similar reasons, features like POS and aspect are not very informative in ordering MEs. Moreover, features like aspect require annotators with not only a clinical background but also some expert knowledge in linguistics, which is not feasible.

6 Conclusions

Representing and reasoning with temporal information in unstructured text is crucial to the field of natural language processing and biomedical informatics. We presented a study on learning to rank medical events. Temporally ordering medical events allows us to induce a partial order of medical events over the patient's history. We noted many differences between learning temporal relations in clinical text and Timebank. The ranking experiments on clinical text yield better performance than classification, whereas the performance is the exact opposite in Timebank. Based on experiments in two very different domains, we demonstrate the need to rethink the resources and methods for temporal relation learning.

Acknowledgments

The project was supported by the NCRR, Grant UL1RR025755, KL2RR025754, and TL1RR025753, is now at the NCATS, Grant 8KL2TR000112-05, 8UL1TR000090-05, 8TL1TR000091-05. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

James F. Allen. 1981. An interval-based representation of temporal knowledge. In *IJCAI*, pages 221–226.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic Acids Research*, 32(suppl 1):D267–D270.

Nathanael Chambers, Shan Wang, and Daniel Jurafsky. 2007. Classifying temporal relations between events. In *ACL*.

A.J. Conger. 1980. Integration and generalization of kappas for multiple raters. In *Psychological Bulletin Vol 88(2)*, pages 322–328.

Thorsten Joachims, Hang Li, Tie-Yan Liu, and ChengXiang Zhai. 2007. Learning to rank for information retrieval (lr4ir 2007). *SIGIR Forum*, 41(2):58–62.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In Bernhard Schölkopf, Christopher John C. Burges, and Alexander J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, pages 169–184. MIT Press.

Thorsten Joachims. 2006. Training linear SVMs in linear time. In *KDD*, pages 217–226.

Mirella Lapata and Alex Lascarides. 2011. Learning sentence-internal temporal relations. *CoRR*, abs/1110.1394.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. In *ACL*.

James Pustejovsky, Jos M. Castao, Robert Ingria, Roser Sauri, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. 2003. TimeML: Robust specification of event and temporal expressions in text. In *New Directions in Question Answering'03*, pages 28–34.

A. Roberts, R. Gaizauskas, M. Hepple, G. Demetriou, Y. Guo, and A. Setzer. 2008. Semantic Annotation of Clinical Text: The CLEF Corpus. In *Proceedings of the LREC 2008 Workshop on Building and Evaluating Resources for Biomedical Text Mining*, pages 19–26.

Guergana K. Savova, Steven Bethard, Will Styler, James Martin, Martha Palmer, James Masanz, and Wayne Ward. 2009. Towards temporal relation discovery from the clinical narrative. *AMIA*.

Marc Verhagen, Robert J. Gaizauskas, Frank Schilder, Mark Hepple, Jessica Moszkowicz, and James Pustejovsky. 2009. The tempeval challenge: identifying temporal relations in text. *Language Resources and Evaluation*, 43(2):161–179.

Li Zhou and George Hripcsak. 2007. Temporal reasoning with medical data - a review with emphasis on medical natural language processing. *Journal of Biomedical Informatics*, pages 183–202.

Li Zhou, Genevieve B. Melton, Simon Parsons, and George Hripcsak. 2006. A temporal constraint structure for extracting temporal information from clinical narrative. *Journal of Biomedical Informatics*, pages 424–439.

A Context-sensitive, Multi-faceted model of Lexico-Conceptual Affect

Tony Veale
Web Science and Technology Division,
KAIST, Daejeon,
South Korea.
Tony.Veale@gmail.com

Abstract

Since we can 'spin' words and concepts to suit our affective needs, context is a major determinant of the perceived affect of a word or concept. We view this re-profiling as a selective emphasis or de-emphasis of the qualities that underpin our shared stereotype of a concept or a word meaning, and construct our model of the affective lexicon accordingly. We show how a large body of affective stereotypes can be acquired from the web, and also show how these are used to create and interpret affective metaphors.

1 Introduction

The builders of affective lexica face the vexing task of distilling the many and varied pragmatic uses of a word or concept into an overall semantic measure of affect. The task is greatly complicated by the fact that in each context of use, speakers may implicitly agree to focus on just a subset of the salient features of a concept, and it is these features that determine contextual affect. Naturally, disagreements arise when speakers do not implicitly arrive at such a consensus, as when people disagree about *hackers*: advocates often focus on qualities that emphasize curiosity or technical virtuosity, while opponents focus on qualities that emphasize criminality and a disregard for the law. In each case, it is the same concept, *Hacker*, that is being described, yet speakers can focus on different qualities to arrive at different affective stances.

Any gross measure of affect (such as e.g., that hackers are *good* or *bad*) must thus be grounded in a nuanced model of the stereotypical properties and behaviors of the underlying word-concept. As different stereotypical qualities are highlighted or de-emphasized in a given context – a particular metaphor, say, might describe *hackers as terrorists* or *hackers as artists* – we need to be able to re-calculate the perceived affect of the word-concept.

This paper presents such a stereotype-grounded model of the affective lexicon. After reviewing the relevant background in section 2, we present the basis of the model in section 3. Here we describe how a large body of feature-rich stereotypes is acquired from the web and from local n-grams. The model is evaluated in section 4. We conclude by showing the utility of the model to that most contextual of NLP phenomena – affective metaphor.

2 Related Work and Ideas

In its simplest form, an affect lexicon assigns an affective score – along one or more dimensions – to each word or sense. For instance, Whissell's (1989) *Dictionary of Affect* (or *DoA*) assigns a trio of scores to each of its 8000+ words to describe three psycholinguistic dimensions: *pleasantness*, *activation* and *imagery*. In the DoA, the lowest pleasantness score of 1.0 is assigned to words like *abnormal* and *ugly*, while the highest, 3.0, is assigned to words like *wedding* and *winning*. Though Whissell's DoA is based on human ratings, Turney (2002) shows how affective valence can be derived from measures of word association in web texts.

Human intuitions are prized in matters of lexical affect. For reliable results on a large-scale, Mohammad & Turney (2010) and Mohammad & Yang (2011) thus used the *Mechanical Turk* to elicit human ratings of the emotional content of words. Ratings were sought along the eight dimensions identified in Plutchik (1980) as primary emotions: *trust , anger*, *anticipation*, *disgust*, *fear*, *joy*, *sadness* and *surprise*. Automated tests were used to exclude unsuitable raters. In all, 24,000+ word-sense pairs were annotated by five different raters.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 75–79,
Jeju, Republic of Korea, 8-14 July 2012.

Liu *et al*. (2003) also present a multidimensional affective model that uses the six basic emotion categories of Ekman (1993) as its dimensions: *happy, sad, angry, fearful, disgusted* and *surprised*. These authors base estimates of affect on the contents of *Open Mind*, a common-sense knowledge-base (Singh, 2002) harvested from contributions of web volunteers. These contents are treated as sentential objects, and a range of NLP models is used to derive affective labels for the subset of contents (~10%) that appear to convey an emotional stance. These labels are then propagated to related concepts (e.g., *excitement* is propagated from *roller-coasters* to *amusement parks*) so that the implicit affect of many other concepts can be determined.

Strapparava and Valitutti (2004) provide a set of affective annotations for a subset of WordNet's synsets in a resource called *Wordnet-affect*. The annotation labels, called *a-labels*, focus on the cognitive dynamics of emotion, allowing one to distinguish e.g. between words that denote an *emotion-eliciting situation* and those than denote an *emotional response*. Esuli and Sebastiani (2006) also build directly on WordNet as their lexical platform, using a semi-supervised learning algorithm to assign a trio of numbers – *positivity*, *negativity* and *neutrality* – to word senses in their newly derived resource, *SentiWordNet*. (*Wordnet-affect* also supports these three dimensions as a-labels, and adds a fourth, *ambiguous*). Esuli & Sebastiani (2007) improve on their affect scores by running a variant of the PageRank algorithm (see also Mihalcea and Tarau, 2004) on the graph structure that tacitly connects word-senses in WordNet to each other via the words used in their textual glosses.

These lexica attempt to capture the affective profile of a word/sense when it is used in its most normative and stereotypical guise, but they do so without an explicit model of stereotypical meaning. Veale & Hao (2007) describe a web-based approach to acquiring such a model. They note that since the simile pattern "as ADJ as DET NOUN" presupposes that NOUN is an exemplar of ADJ*ness*, it follows that ADJ must be a highly salient property of NOUN. Veale & Hao harvested tens of thousands of instances of this pattern from the Web, to extract sets of adjectival properties for thousands of commonplace nouns. They show that if one estimates the pleasantness of a term like *snake* or *artist* as a weighted average of the pleasantness of its properties (like *sneaky* or *creative*) in a resource like Whissell's DoA, then the estimated scores show a reliable correlation with the DoA's own scores. It thus makes computational sense to calculate the affect of a word-concept as a function of the affect of its most salient properties. Veale (2011) later built on this work to show how a property-rich stereotypical representation could be used for non-literal matching and retrieval of creative texts, such as metaphors and analogies.

Both Liu et al. (2003) and Veale & Hao (2010) argue for the importance of common-sense knowledge in the determination of affect. We incorporate ideas from both here, while choosing to build mainly on the latter, to construct a nuanced, two-level model of the affective lexicon.

3 An Affective Lexicon of Stereotypes

We construct the stereotype-based lexicon in two stages. For the first layer, a large collection of stereotypical descriptions is harvested from the web. As in Liu *et al*. (2003), our goal is to acquire a lightweight common-sense representation of many everyday concepts. For the second layer, we link these common-sense qualities in a *support graph* that captures how they mutually support each other in their co-description of a stereotypical idea. From this graph we can estimate pleasantness and unpleasantness valence scores for each property and behavior, and for the stereotypes that exhibit them.

Expanding on the approach in Veale (2011), we use two kinds of query for harvesting stereotypes from the web. The first, "as ADJ as a NOUN", acquires typical adjectival properties for noun concepts; the second, "VERB+*ing* like a NOUN" and "VERB+*ed* like a NOUN", acquires typical verb behaviors. Rather than use a wildcard * in both positions (ADJ and NOUN, or VERB and NOUN), which gives limited results with a search engine like Google, we generate fully instantiated similes from hypotheses generated via the Google n-grams (Brants & Franz, 2006). Thus, from the 3-gram "a drooling zombie" we generate the query "drooling *like* a zombie", and from the 3-gram "a mindless zombie" we generate "*as* mindless *as* a zombie".

Only those queries that retrieve one or more Web documents via the Google API indicate the most promising associations. This still gives us over 250,000 web-validated simile associations for our stereotypical model, and we filter these manually, to ensure that the lexicon is both reusable and

of the highest quality. We obtain rich descriptions for many stereotypical ideas, such as *Baby*, which is described via 163 typical properties and behaviors like *crying*, *drooling* and *guileless*. After this phase, the lexicon maps each of 9,479 stereotypes to a mix of 7,898 properties and behaviors.

We construct the second level of the lexicon by automatically linking these properties and behaviors to each other in a support graph. The intuition here is that properties which reinforce each other in a single description (e.g. "as *lush and green* as a jungle" or "as *hot and humid* as a sauna") are more likely to have a similar affect than properties which do not support each other. We first gather all Google 3-grams in which a pair of stereotypical properties or behaviors X and Y are linked via coordination, as in "*hot and humid*" or "*kicking and screaming*". A bidirectional link between X and Y is added to the support graph if one or more stereotypes in the lexicon contain both X and Y. If this is not so, we also ask whether both descriptors ever reinforce each other in Web similes, by posing the web query "*as X and Y as*". If this query has non-zero hits, we still add a link between X and Y.

Let **N** denote this support graph, and N(p) denote the set of neighboring terms to p, that is, the set of properties and behaviors that can mutually support a property p. Since every edge in **N** represents an affective context, we can estimate the likelihood that p is ever used in a positive or negative context if we know the positive or negative affect of enough members of N(p). So if we label enough vertices of **N** with + / – labels, we can interpolate a positive/negative affect for all vertices p in **N**.

We thus build a reference set **-R** of typically negative words, and a set **+R** of typically positive words. Given a few seed members of **-R** (such as *sad*, *evil*, etc.) and a few seed members of **+R** (such as *happy*, *wonderful*, etc.), we find many other candidates to add to **+R** and **-R** by considering neighbors of these seeds in **N**. After just three iterations, **+R** and **-R** contain ~2000 words each.

For a property p, we define $N^+(p)$ and $N^-(p)$ as

$$(1) \quad N^+(p) = N(p) \cap \mathbf{+R}$$

$$(2) \quad N^-(p) = N(p) \cap \mathbf{-R}$$

We assign pos/neg valence scores to each property p by interpolating from reference values to their neighbors in **N**. Unlike that of Takamura *et al*. (2005), the approach is non-iterative and involves no feedback between the nodes of **N,** and thus, no inter-dependence between adjacent affect scores:

$$(3) \quad \text{pos}(p) = \frac{|N^+(p)|}{|N^+(p) \cup N^-(p)|}$$

$$(4) \quad \text{neg}(p) = 1 - \text{pos}(p)$$

If a term S denotes a stereotypical idea and is described via a set of typical properties and behaviors *typical(S)* in the lexicon, then:

$$(5) \quad \text{pos}(S) = \frac{\sum_{p \in \text{typical}(S)} \text{pos}(p)}{|\text{typical}(S)|}$$

$$(6) \quad \text{neg}(S) = 1 - \text{pos}(S)$$

Thus, (5) and (6) calculate the mean affect of the properties and behaviors of S, as represented via *typical(S)*. We can now use (3) and (4) to separate *typical(S)* into those elements that are more negative than positive (putting an unpleasant spin on S in context) and those that are more positive than negative (putting a pleasant spin on S in context):

$$(7) \quad \text{posTypical}(S) = \{p \mid p \in \text{typical}(S) \wedge \text{pos}(p) > 0.5\}$$

$$(8) \quad \text{negTypical}(S) = \{p \mid p \in \text{typical}(S) \wedge \text{neg}(p) > 0.5\}$$

4 Empirical Evaluation

In the process of populating **+R** and **-R**, we identify a reference set of 478 positive stereotype nouns (such as *saint* and *hero*) and 677 negative stereotype nouns (such as *tyrant* and *monster*). We can use these reference stereotypes to test the effectiveness of (5) and (6), and thus, indirectly, of (3) and (4) and of the affective lexicon itself. Thus, we find that **96.7%** of the stereotypes in **+R** are correctly assigned a positivity score greater than 0.5 (*pos(S) > neg(S)*) by (5), while **96.2%** of the stereotypes in **-R** are correctly assigned a negativity score greater than 0.5 (*neg(S) > pos(S)*) by (6).

We can also use **+R** and **-R** as a gold standard for evaluating the separation of *typical*(S) into distinct positive and negative subsets *posTypical(S)* and *negTypical(S)* via (7) and (8). The lexicon contains 6,230 stereotypes with at least one property in **+R**∪**-R**. On average, **+R**∪**-R** contains 6.51 of the properties of each of these stereotypes, where, on average, 2.95 are in **+R** while 3.56 are in **-R**.

In a perfect separation, (7) should yield a positive subset that contains only those properties in

typical(*S*)∩**+R,** while (8) should yield a negative subset that contains only those in *typical*(*S*)∩**-R**.

Macro Averages **(6230 stereotypes)**	***Positive*** **properties**	***Negative*** **properties**
Precision	.962	.98
Recall	.975	.958
F-Score	.968	.968

Table 1. Average P/R/F1 scores for the affective retrieval of +/- properties from 6,230 stereotypes.

Viewing the problem as a retrieval task then, in which (7) and (8) are used to retrieve distinct positive and negative property sets for a stereotype *S*, we report the encouraging results of Table 1 above.

5 Re-shaping Affect in Figurative Contexts

The Google n-grams are a rich source of affective metaphors of the form *Target is Source*, such as "politicians are crooks", "Apple is a cult", "racism is a disease" and "Steve Jobs is a god". Let *src(T)* denote the set of stereotypes that are commonly used to describe T, where commonality is defined as the presence of the corresponding copula metaphor in the Google n-grams. Thus, for example:

src(racism) = {*problem, disease, poison, sin, crime, ideology, weapon, ...*}

src(Hitler) = {*monster, criminal, tyrant, idiot, madman, vegetarian, racist, ...*}

Let *srcTypical*(T) denote the aggregation of all properties ascribable to T via metaphors in *src*(T):

$$(9)\quad srcTypical(T) = \bigcup_{M \in src(T)} typical(M)$$

We can also use the *posTypical* and *negTypical* variants in (7) and (8) to focus only on metaphors that project positive or negative qualities onto T.

In effect, (9) provides a feature representation for a topic T as viewed through the prism of metaphor. This is useful when the source *S* in the metaphor *T is S* is not a known stereotype in the lexicon, as happens e.g. in *Apple is Scientology*. We can also estimate whether a given term *S* is more positive than negative by taking the average pos/neg valence of *src*(S). Such estimates are 87% correct when evaluated using **+R** and **-R** examples.

The properties and behaviors that are contextually relevant to the interpretation of *T is S* are given by

$$(10)\quad salient(T,S) = |srcTypical(T) \cup typical(T)| \cap |srcTypical(S) \cup typical(S)|$$

In the context of *T is S*, the figurative perspective $M \in src(S) \cup src(T) \cup \{S\}$ is deemed apt for T if:

$$(11)\quad apt(M, T,S) = |salient(T,S) \cap typical(M)| > 0$$

and the degree to which M is apt for T is given by:

$$(12)\quad aptness(M,T,S) = \frac{|salient(T, S) \cap typical(M)|}{|typical(M)|}$$

We can construct an interpretation for *T is S* by considering not just {S}, but the stereotypes in *src*(T) that are apt for T in the context of *T is S*, as well as the stereotypes that are commonly used to describe S – that is, *src*(S) – that are also apt for T:

$$(13)\quad interpretation(T, S) = \{M | M \in src(T) \cup src(S) \cup \{S\} \wedge apt(M, T, S)\}$$

The elements $\{M_i\}$ of *interpretation*(T, S) can now be sorted by $aptness(M_i\ T, S)$ to produce a ranked list of interpretations (M_1, M_2 … M_n). For any interpretation M, the salient features of M are thus:

$$(14)\quad salient(M, T,S) = typical(M) \cap salient(T,S)$$

So *interpretation*(T, S) is an expansion of the affective metaphor *T is S* that includes the common metaphors that are consistent with T *qua* S. For instance, "*Google is -Microsoft*" (where - indicates a negative spin) produces *{monopoly, threat, bully, giant, dinosaur, demon, ...}*. For each M_i in *interpretation*(T, S), *salient*(M_i, *T, S*) is an expansion of M_i that includes all of the qualities that are apt for T *qua* M_i (e.g. *threatening*, *sprawling*, *evil*, etc.).

6 Concluding Remarks

Metaphor is the perfect tool for influencing the perceived affect of words and concepts in context. The web application *Metaphor Magnet* provides a proof-of-concept demonstration of this re-shaping process at work, using the stereotype lexicon of §3, the selective highlighting of (7)–(8), and the model of metaphor in (9)–(14). It can be accessed at:

http://*boundinanutshell.com/metaphor-magnet*

Acknowledgements

This research was supported by the WCU (World Class University) program under the National Research Foundation of Korea, and funded by the Ministry of Education, Science and Technology of Korea (Project No: R31-30007).

References

Thorsten Brants and Alex Franz. 2006. *Web 1T 5-gram Version 1*. Linguistic Data Consortium.

Paul Ekman. 1993. Facial expression of emotion. American Psychologist, 48:384-392.

Andrea Esuli and Fabrizio Sebastiani. 2006. SentiWordNet: A publicly available lexical resource for opinion mining. Proc. of LREC-2006, the 5th Conference on Language Resources and Evaluation, 417-422.

Andrea Esuli and Fabrizio Sebastiani. 2007. PageRanking WordNet Synsets: An application to opinion mining. Proc. of ACL-2007, the 45th Annual Meeting of the Association for Computational Linguistics.

Hugo Liu, Henry Lieberman, and Ted Selker. 2003. A Model of Textual Affect Sensing Using Real-World Knowledge. Proceedings of the 8th international conference on Intelligent user interfaces, pp. 125-132.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order to Texts. Proceedings of EMNLP-04, the 2004 Conference on Empirical Methods in Natural Language Processing.

Saif F. Mohammad and Peter D. Turney. 2010. Emotions evoked by common words and phrases: Using Mechanical Turk to create an emotional lexicon. Proceedings of the NAACL-HLT 2010 workshop on Computational Approaches to Analysis and Generation of Emotion in Text. Los Angeles, CA.

Saif F. Mohammad and Tony Yang. 2011. Tracking sentiment in mail: how genders differ on emotional axes. Proceedings of the ACL 2011 WASSA workshop on Computational Approaches to Subjectivity and Sentiment Analysis, Portland, Oregon.

Robert Plutchik. 1980. A general psycho-evolutionary theory of emotion. Emotion: Theory, research and experience, 2(1-2):1-135.

Push Singh. 2002. The public acquisition of commonsense knowledge. Proceedings of AAAI Spring Symposium on Acquiring (and Using) Linguistic (and World) Knowledge for Information Access. Palo Alto, CA.

Carlo Strapparava and Alessandro Valitutti. 2004. Wordnet-affect: an affective extension of Wordnet. Proceedings of LREC-2004, the 4th International Conference on Language Resources and Evaluation, Lisbon, Portugal.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2005. Extracting semantic orientation of words using spin model. Proceedings of the 43rd Annual Meeting of the ACL, 133–140.

Turney, P. D. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In Proceedings of ACL-2002, the 40th Annual Meeting of the Association for Computational Linguistics, pp. 417–424. June 2002.

Veale, T. and Hao, Y. Making Lexical Ontologies Functional and Context-Sensitive. Proceedings of ACL-2007, the 45th Annual Meeting of the Association of Computational Linguistics, pp. 57–64. June 2007.

Veale, T. and Hao, Y. Detecting Ironic Intent in Creative Comparisons. Proceedings of ECAI'2010, the 19th European Conference on Artificial Intelligence, Lisbon. August 2010.

Veale, T. Creative Language Retrieval: A Robust Hybrid of Information Retrieval and Linguistic Creativity. Proceedings of ACL'2011, the 49th Annual Meeting of the Association of Computational Linguistics. June 2011.

Whissell, C. The dictionary of affect in language. In R. Plutchik and H. Kellerman (Eds.) Emotion: Theory and research. Harcourt Brace, pp. 113-131. 1989.

Decoding Running Key Ciphers

Sravana Reddy*
Department of Computer Science
The University of Chicago
1100 E. 58th Street
Chicago, IL 60637, USA
sravana@cs.uchicago.edu

Kevin Knight
Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292, USA
knight@isi.edu

Abstract

There has been recent interest in the problem of decoding letter substitution ciphers using techniques inspired by natural language processing. We consider a different type of classical encoding scheme known as the *running key cipher*, and propose a search solution using Gibbs sampling with a word language model. We evaluate our method on synthetic ciphertexts of different lengths, and find that it outperforms previous work that employs Viterbi decoding with character-based models.

1 Introduction

The running key cipher is an encoding scheme that uses a secret key R that is typically a string of words, usually taken from a book or other text that is agreed upon by the sender and receiver. When sending a plaintext message P, the sender truncates R to the length of the plaintext. The scheme also relies on a substitution function f, which is usually publicly known, that maps a plaintext letter p and key letter r to a unique ciphertext letter c. The most common choice for f is the *tabula recta*, where $c = (p + r) \mod 26$ for letters in the English alphabet, with A $= 0$, B $= 1$, and so on.

To encode a plaintext with a running key, the spaces in the plaintext and the key are removed, and for every $0 \leq i < |P|$, the ciphertext letter at position i is computed to be $C_i \leftarrow f(P_i, R_i)$. Figure 1 shows an example encoding using the *tabula recta*.

For a given ciphertext and known f, the plaintext uniquely determines the running key and vice versa. Since we know that the plaintext and running key are both drawn from natural language, our objective function for the solution plaintext under some language model is:

$$\hat{P} = \arg\max_{P} \log \Pr(P) \Pr(R_{P,C}) \qquad (1)$$

where the running key $R_{P,C}$ is the key that corresponds to plaintext P and ciphertext C.

Note that if $R_{P,C}$ is a perfectly random sequence of letters, this scheme is effectively a 'one-time pad', which is provably unbreakable (Shannon, 1949). The knowledge that both the plaintext and the key are natural language strings is important in breaking a running key cipher.

The letter-frequency distribution of running key ciphertexts is notably flatter than than the plaintext distribution, unlike substitution ciphers where the frequency profile remains unchanged, modulo letter substitutions. However, the ciphertext letter distribution is not uniform; there are peaks corresponding to letters (like I) that are formed by high-frequency plaintext/key pairs (like E and E).

2 Related Work

2.1 Running Key Ciphers

Bauer and Tate (2002) use letter n-grams (without smoothing) up to order 6 to find the most probable plaintext/key character pair at each position in the ciphertext. They test their method on 1000-character ciphertexts produced from plaintexts and keys extracted from Project Gutenberg. Their accuracies range from 28.9% to 33.5%, where accuracy is measured as the percentage of correctly decoded char-

*Research conducted while the author was visiting ISI.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 80–84,
Jeju, Republic of Korea, 8-14 July 2012.

Figure 1: Example of a running key cipher. Note that key is truncated to the length of the plaintext.
Plaintext – linguistics is fun, *Running Key* – colorless green ideas, *tabula recta* substitution where $C_i \leftarrow (P_i + R_i) \mod 26$

Plaintext:	L	I	N	G	U	I	S	T	I	C	S	I	S	F	U	N
Running Key:	C	O	L	O	R	L	E	S	S	G	R	E	E	N	I	D
Ciphertext:	N	W	Y	U	L	T	W	L	A	I	J	M	W	S	C	Q

acters. Such figures are too low to produce readable plaintexts, especially if the decoded regions are not contiguous. Griffing (2006) uses Viterbi decoding and letter 6-grams to improve on the above result, achieving a median 87% accuracy over several 1000-character ciphertexts. A key shortcoming of this work is that it requires searching through about 26^5 states at each position in the ciphertext.

2.2 Letter Substitution Ciphers

Previous work in decipherment of classical ciphers has mainly focused on letter substitution. These ciphers use a substitution table as the secret key. The ciphertext is generated by substituting each letter of the plaintext according to the substitution table. The table may be homophonic; that is, a single plaintext letter could map to more than one possible ciphertext letter. Just as in running key ciphers, spaces in the plaintext are usually removed before encoding.

Proposed decipherment solutions for letter substitution ciphers include techniques that use expectation maximization (Ravi and Knight, 2008), genetic algorithms (Oranchak, 2008), integer programming (Ravi and Knight, 2009), A* decoding (Corlett and Penn, 2010), and Bayesian learning with Dirichlet processes (Ravi and Knight, 2011).

2.3 Vigenère Ciphers

A scheme similar to the running key cipher is the Vigenère cipher, also known as the periodic key cipher. Instead of a single long string spanning the length of the plaintext, the key is a short string – usually but not always a single word or phrase – repeated to the length of the plaintext. Figure 2 shows an example Vigenère cipher encoding. This cipher is less secure than the running key, since the short length of the key vastly reduces the size of the search space, and the periodic repetition of the key leaks information.

Recent work on decoding periodic key ciphers perform Viterbi search on the key using letter n-gram models (Olsen et al., 2011), with the assumption that the length of the key is known. If unknown, the key length can be inferred using the Kasiski Test (Kasiski, 1863) which takes advantage of repeated plaintext/key character pairs.

3 Solution with Gibbs Sampling

In this paper, we describe a search algorithm that uses Gibbs Sampling to break a running key cipher.

3.1 Choice of Language Model

The main advantage of a sampling-based approach over Viterbi decoding is that it allows us to seamlessly use word-based language models. Lower order letter n-grams may fail to decipher most ciphertexts even with perfect search, since an incorrect plaintext and key could have higher likelihood under a weak language model than the actual message.

3.2 Blocked Sampling

One possible approach is to sample a plaintext letter at each position in the ciphertext. The limitation of such a sampler for the running key problem is that is extremely slow to mix, especially for longer ciphertexts: we found that in practice, it does not usually converge to the optimal solution in a reasonable number of iterations even with simulated annealing. We therefore propose a blocked sampling algorithm that samples *words* rather than letters in the plaintext, as follows:

1. Initialize randomly $P := p_1 p_2 \ldots p_{|C|}$, fix R as the key that corresponds to P, C
2. Repeat for some number of iterations
   (a) Sample spaces (word boundaries) in P according to $\Pr(P)$
   (b) Sample spaces in R according to $\Pr(R)$
   (c) Sample each word in P according to $\Pr(P)\Pr(R)$, updating R along with P
   (d) Sample each word in R according to $\Pr(P)\Pr(R)$, updating P along with R

Figure 2: Example of a Vigenère cipher cipher, with a 5-letter periodic key, repeated to the length of the plaintext. *Plaintext* – linguistics is fun, *Periodic Key* – green, *tabula recta* substitution.

Plaintext:	L	I	N	G	U	I	S	T	I	C	S	I	S	F	U	N
Running Key:	G	R	E	E	N	G	R	E	E	N	G	R	E	E	N	G
Ciphertext:	R	Z	R	K	H	O	J	X	M	P	Y	Z	W	J	H	T

3. Remove spaces and return P, R

Note that every time a word in P is sampled, it induces a change in R that may not be a word or a sequence of words, and vice versa. Sampling word boundaries will also produce hypotheses containing non-words. For this reason, we use a word trigram model linearly interpolated with letter trigrams (including the space character).[1] The interpolation mainly serves to smooth the search space, with the added benefit of accounting for out-of-vocabulary, misspelled, or truncated words in the actual plaintext or key. Table 1 shows an example of one sampling iteration on the ciphertext shown in Figure 1.

Table 1: First sampling iteration on the ciphertext NWYULTWLAIJMWSCQ

Generate P, R with letter trigrams	P: WERGATERYBVIEDOW R: RSHOLASUCHOESPOU
Sample spaces in P	P: WERGAT ER YB VIEDOW
Sample spaces in R	R: RS HOLASUCHOES POU
Sample words in P	P: ADJUST AN MY WILLOW R: NT PATAWYOKNEL HOU
Sample words in R	P: NEWNXI ST HE SYLACT R: AS CHOLESTEROL SAX

4 Experiments

4.1 Data

We randomly select passages from the Project Gutenberg and Wall Street Journal Corpus extracts that are included in the NLTK toolkit (Bird et al., 2009). The passages are used as plaintext and key pairs, and combined to generate synthetic ciphertext data. Unlike previous works which used constant-length ciphertexts, we study the effect of message length on decipherment by varying the ciphertext length (10, 100, and 1000 characters).

Our language model is an interpolation of word trigrams and letter trigrams trained on the Brown Corpus (Nelson and Kucera, 1979), with Kneser-Ney smoothing. We fixed the word language model interpolation weight to $\lambda = 0.7$.

[1] $\Pr(P) = \lambda \Pr(P|\text{word LM}) + (1 - \lambda) \Pr(P|\text{letter LM})$, and similarly for $\Pr(R)$.

4.2 Baseline and Evaluation

For comparison with the previous work, we re-implement Viterbi decoding over letter 6-grams (Griffing, 2006) trained on the Brown Corpus. In addition to decipherment accuracy, we compare the running time in seconds of the two algorithms. Both decipherment programs were implemented in Python and run on the same machines. The Gibbs Sampler was run for 10000 iterations.

As in the Griffing (2006) paper, since the plaintext and running key are interchangeable, we measure the accuracy of a hypothesized solution against the reference as the max of the accuracy between the hypothesized plaintext and the reference plaintext, and the hypothesized plaintext and the reference key.

4.3 Results

Table 2 shows the average decipherment accuracy of our algorithm and the baseline on each dataset. Also shown is the number of times that the Gibbs Sampling search failed – that is, when the algorithm did not hypothesize a solution that had a probability at least as high as the reference within 10000 iterations.

It is clear that the Gibbs Sampler is orders of magnitude faster than Viterbi decoding. Performance on the short (length 10) ciphertexts is poor under both algorithms. This is expected, since the degree of message uncertainty, or *message equivocation* as defined by Shannon, is high for short ciphertexts: there are several possible plaintexts and keys besides the original that are likely under an English language model. Consider the ciphertext WAEEXFPROV which was generated by the plaintext segment ON A REFEREN and key INENTAL AKI. The algorithm hypothesizes that the plaintext is THE STRAND S and key DTAME OPELD, which both receive high language model probability.

Table 2: Decipherment accuracy (proportion of correctly deciphered characters). Plaintext and key sources for the ciphertext test data were extracted by starting at random points in the corpora, and selecting the following n characters.

Length of ciphertext	Plaintext and key source	# Cipher-texts	Average Accuracy		Avg. running time (sec)		# Failed Gibbs searches
			Viterbi	**Gibbs**	Viterbi	**Gibbs**	
10	Project Gutenberg	100	14%	**17%**	1005	**47**	5
	Wall Street Journal	100	10%	**26%**	986	**38**	2
100	Project Gutenberg	100	27%	**42%**	10212	**236**	19
	Wall Street Journal	100	22%	**58%**	10433	**217**	12
1000	Project Gutenberg	100	63%	**88%**	112489	**964**	32
	Wall Street Journal	100	60%	**93%**	117303	**1025**	25

Table 3: Substitution function parameterized by the keyword, CIPHER. $f(p, r)$ is the entry in the row corresponding to p and the column corresponding to r.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
A	C	I	P	H	E	R	A	B	D	F	G	J	K	L	M	N	O	Q	S	T	U	V	W	X	Y	Z
B	I	P	H	E	R	A	B	D	F	G	J	K	L	M	N	O	Q	S	T	U	V	W	X	Y	Z	C
C	P	H	E	R	A	B	D	F	G	J	K	L	M	N	O	Q	S	T	U	V	W	X	Y	Z	C	I
...																										

However, on the long ciphertexts, our algorithm gets close to perfect decipherment, surpassing the Viterbi algorithm by a large margin.[2] Accuracies on the Wall Street Journal ciphertexts are higher than on the Gutenberg ciphertexts for our algorithm, which may be because the latter is more divergent from the Brown Corpus language model.

5 Future Work

5.1 Unknown substitution functions

Some running key ciphers also use a secret substitution function f rather than the *tabula recta* or another known function. In typical cases, these functions are not arbitrary, but are parameterized by a secret keyword that mutates the *tabula recta* table. For example, the function with the keyword CIPHER would be the substitution table shown in Table 3. Decoding a running key ciphertext under a latent substitution function is an open line of research. One possibility is to extend our approach by sampling the keyword or function in addition to the plaintext.

5.2 Exact search

Since some the errors in Gibbs Sampling decipherment are due to search failures, a natural extension of this work would be to adapt Viterbi search or other exact decoding algorithms like A* to use word-level language models. A naive implementation of Viterbi word-based decoding results in computationally inefficient search spaces for large vocabularies, so more sophisticated methods or heuristics will be required.

5.3 Analysis of Running Key Decipherment

While there has been theoretical and empirical analysis of the security of letter substitution ciphers of various lengths under different language models (Shannon, 1949; Ravi and Knight, 2008), there has been no similar exposition of running key ciphers, which we reserve for future work.

6 Conclusion

We propose a decipherment algorithm for running key ciphers that uses Blocked Gibbs Sampling and word-based language models, which shows significant speed and accuracy improvements over previous research into this problem.

Acknowledgments

We would like to thank Sujith Ravi for initial experiments using Gibbs sampling, and the anonymous reviewers. This research was supported in part by NSF grant 0904684.

[2]The accuracies that we found for Viterbi decoding are lower than those reported by Griffing (2006), which might be because they use an in-domain language model.

References

Craig Bauer and Christian Tate. 2002. A statistical attack on the running key cipher. *Cryptologia*, 26(4).

Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Eric Corlett and Gerald Penn. 2010. An exact A* method of deciphering letter-substitution ciphers. In *Proceedings of ACL*.

Alexander Griffing. 2006. Solving the running key cipher with the Viterbi algorithm. *Cryptologia*, 30(4).

Friedrich Kasiski. 1863. *Die Geheimschriften und die Dechiffrir-Kunst*. E. S. Mittler und Sohn.

Francis Nelson and Henry Kucera. 1979. *The Brown Corpus: A Standard Corpus of Present-Day Edited American English*. Brown University.

Peder Olsen, John Hershey, Steven Rennie, and Vaibhava Goel. 2011. A speech recognition solution to an ancient cryptography problem. Technical Report RC25109 (W1102-005), IBM Research.

David Oranchak. 2008. Evolutionary algorithm for decryption of monoalphabetic homophonic substitution ciphers encoded as constraint satisfaction problems. In *Proceedings of the Conference on Genetic and Evolutionary Computation*.

Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of EMNLP*.

Sujith Ravi and Kevin Knight. 2009. Attacking letter substitution ciphers with integer programming. *Cryptologia*, 33(4).

Sujith Ravi and Kevin Knight. 2011. Bayesian inference for Zodiac and other homophonic ciphers. In *Proceedings of ACL*.

Claude Shannon. 1949. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4).

Using Rejuvenation to Improve Particle Filtering for Bayesian Word Segmentation

Benjamin Börschinger[*†]
benjamin.borschinger@mq.edu.au

Mark Johnson[*]
mark.johnson@mq.edu.au

[*]Department of Computing
Macquarie University
Sydney, Australia

[†]Department of Computational Linguistics
Heidelberg University
Heidelberg, Germany

Abstract

We present a novel extension to a recently proposed incremental learning algorithm for the word segmentation problem originally introduced in Goldwater (2006). By adding *rejuvenation* to a particle filter, we are able to considerably improve its performance, both in terms of finding higher probability and higher accuracy solutions.

1 Introduction

The goal of word segmentation is to segment a stream of segments, e.g. characters or phonemes, into words. For example, given the sequence "youwanttoseethebook", the goal is to recover the segmented string "you want to see the book". The models introduced in Goldwater (2006) solve this problem in a fully unsupervised way by defining a generative process for word sequences, making use of the Dirichlet Process (DP) prior.

Until recently, the only inference algorithm applied to these models were batch Markov Chain Monte Carlo (MCMC) sampling algorithms. Börschinger and Johnson (2011) proposed a strictly incremental particle filter algorithm that, however, performed considerably worse than the standard batch algorithms, in particular for the Bigram model. We extend that algorithm by adding *rejuvenation* steps and show that this leads to considerable improvements, thus strengthening the case for particle filters as another tool for Bayesian inference in computational linguistics.

The rest of the paper is structured as follows. Sections 2 and 3 provide the relevant background about word segmentation and previous work. Section 4 describes our algorithm. Section 5 reports on an experimental evaluation of our algorithm, and section 6 concludes and suggests possible directions for future research.

2 Model description

The *Unigram* model assumes that words in a sequence are generated independently whereas the *Bigram* model models dependencies between adjacent words. This has been shown by Goldwater (2006) to markedly improve segmentation performance. We perform experiments on both models but, for reasons of space, only give an overview of the Unigram model, referring the reader to the original papers for more detailed descriptions. (Goldwater, 2006; Goldwater et al., 2009)

A sequence of words or utterance is generated by making independent draws from a discrete distribution over words, G. As neither the actual "true" words nor their number is known in advance, G is modelled as a draw from a DP. A DP is parametrized by a base distribution P_0 and a concentration parameter α. Here, P_0 assigns a probability to every possible word, i.e. sequence of segments, and α controls the sparsity of G; the smaller α, the sparser G tends to be.

To computationally cope with the unbounded nature of draws from a DP, they can be "integrated out", yielding the Chinese Restaurant Process (CRP), an infinitely exchangeable conditional predictive distribution. The CRP also provides an intuitive generative story for the observed data. Each generated word token corresponds to a customer sit-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 85–89,
Jeju, Republic of Korea, 8-14 July 2012.

ting at one of the unboundedly many tables in an imaginary Chinese restaurant. Customers choose their seats sequentially, and they sit either at an already occupied or a new table. The former happens with probability proportional to the number of customers already sitting at a table and corresponds to generating one more token of the word type all customers at a table instantiate. The latter happens with probability proportional to α and corresponds to generating a token by sampling from the base distribution, thus also determining the type for all potential future customers at the new table.

Given this generative process, word segmentation can be cast as a probabilistic inference problem. For a fixed input, in our case a sequence of phonemes, our goal is to determine the posterior distribution over segmentations. This is usually infeasible to do exactly, leading to the use of approximate inference methods.

3 Previous Work

The "standard" inference algorithms for the Unigram and Bigram model are MCMC samplers that are batch algorithms making multiple iterations over the data to non-deterministically explore the state space of possible segmentations. If an MCMC algorithm runs long enough, the probability of it visiting any specific segmentation is the probability of that segmentation under the target posterior distribution, here, the distribution over segmentations given the observed data.

The MCMC algorithm of Goldwater et al. (2009) is a *Gibbs* sampler that makes very small moves through the state space by changing individual word boundaries one at a time. An alternative MCMC algorithm that samples segmentations for entire utterances was proposed by Mochihashi et al. (2009). Below, we correct a minor error in the algorithm, recasting it as a Metropolis-within-Gibbs sampler.

Moving beyond MCMC algorithms, Pearl et al. (2010) describe an algorithm that can be seen as a degenerate limiting case of a particle filter with only one particle. Their *Dynamic Programming Sampling* algorithm makes a single pass through the data, processing one utterance at a time by sampling a segmentation given the choices made for all previous utterances. While their algorithm comes with no guarantee that it converges on the intended posterior distribution, Börschinger and Johnson (2011) showed how to construct a particle filter that is asymptotically correct, although experiments suggested that the number of particles required for good performance is impractically large.

This paper shows how their algorithm can be improved by adding *rejuvenation steps*, which we will describe in the next section.

4 A Particle Filter with Rejuvenation

The core idea of a *particle filter* is to sequentially approximate a target posterior distribution P by N weighted point samples or "particles". Each particle is updated one observation at a time, exploiting the insight that Bayes' Theorem can be applied recursively, as illustratively shown for the case of calculating the posterior probability of a hypothesis H given two observations O_1 and O_2:

$$P(H|O_1) \propto P(O_1|H)P(H) \quad (1)$$

$$P(H|O_1, O_2) \propto P(O_2|H)P(H|O_1) \quad (2)$$

If the observations are conditionally independent given the hypothesis, one can simply take the posterior at time step t as the prior for the posterior update at time step $t + 1$.

Here, each particle corresponds to a specific segmentation of the data observed so far, or more precisely, the specific CRP seating of word tokens in this segmentation; we refer to this as its *history*. Its weight indicates how well a particle is supported by the data, and each observation corresponds to an unsegmented utterance. With this, the basic particle filter algorithm can be described as follows: Begin with N "empty" particles. To get the particles at time $t+1$ from the particles at time t, update each particle using the observation at time $t+1$ as follows: sample a segmentation for this observation, given the particle's history, then add the words in this segmentation to that history. After each particle has been updated, their weights are adjusted to reflect how well they are now supported by the observations. The set of updated and reweighted particles constitutes the approximation of the posterior at time $t + 1$.

To overcome the problem of degeneracy (the situation where only very few particles have non-negligible weights), Börschinger and Johnson use

resampling; basically, high-probability particles are permitted to have multiple descendants that can replace low-probability particles. For reasons of space, we refer the reader to Börschinger and Johnson (2011) for the details of these steps.

While necessary to address the degeneracy problem, resampling leads to a *loss of sample diversity*; very quickly, almost all particles have an identical history, descending from only a small number of (previously) high probability particles. With a strict online learning constraint, this can only be counteracted by using an extremely large number of particles. An alternative strategy which we explore here is to use *rejuvenation*; the core idea is to restore sample diversity after each resampling step by performing MCMC resampling steps on each particle's history, thus leading to particles with different histories in each generation, even if they all have the same parent. (e.g., Canini et al. (2009)) This makes it necessary to store previously processed observations and thus no longer qualifies as online learning in a strict sense, but it still yields an incremental algorithm that learns as the observations arrive sequentially, instead of delaying learning until all observations are available.

In our setting, rejuvenation works as follows. After each resampling step, for each particle the algorithm performs a fixed number of the following rejuvenation steps:

1. randomly choose a previously observed utterance
2. resample the segmentation for this utterance and update the particle accordingly

For the resampling step, we use Mochihashi et al. (2009)'s algorithm to efficiently sample segmentations for an unsegmented utterance o, given a sequence of n previously observed words $W_{1:n}$. As the CRP is exchangeable, during resampling we can treat every utterance as if it were the last, making it possible to use this algorithm for any utterance, irrespective of its actual position in the data. Crucially, however, the distribution over segmentations that this algorithm samples from is not the true posterior distribution $P(\cdot|o, \alpha, W_{1:n})$ as defined by the CRP, but a slightly different *proposal* distribution $Q(\cdot|o, \alpha, W_{1:n})$ that does not take into account the intra-sentential word dependencies for a segmentation of o. It is precisely because we ignore these dependencies that an efficient dynamic programming algorithm is possible, but because Q is different from the target conditional distribution P, our algorithm that uses Q instead of P needs to correct for this. In a particle filter, this is done when the particle weights are calculated (Börschinger and Johnson, 2011). For an MCMC algorithm or our rejuvenation step, a *Metropolis-Hastings accept/reject* step is required, as described in detail by Johnson et al. (2007) in the context of grammatical inference.[1]

In our case, during rejuvenation an utterance u with current segmentation s is reanalyzed as follows:

- remove all the words contained in s from the particle's current state L, yielding state $L*$
- sample a proposal segmentation s' for u from $Q(\cdot|u, L*, \alpha)$, using Mochihashi et al. (2009)'s dynamic programming algorithm
- calculate $m = \min\{1, \frac{P(s'|L*,\alpha)Q(s|L*,\alpha)}{P(s|L*,\alpha)Q(s'|L*,\alpha)}\}$
- with probability m, accept the new sample and update $L*$ accordingly, else keep the original segmentation and set the particle's state back to L

This completes the description of our extension to the algorithm. The remainder of the paper empirically evaluates the particle filter with rejuvenation.

5 Experiments

We compare the performance of a batch Metropolis-Hastings sampler for the Unigram and Bigram model with that of particle filter learners both with and without rejuvenation, as described in the previous section. For the batch samplers, we use simulated annealing to facilitate the finding of high probability solutions, and for the particle filters, we compare the performance of a 'degenerate' 1-particle learner with a 16-particle learner in the rejuvenation setting.

To get an impression of the contribution of particle number and rejuvenation steps, we compare

[1]Because Mochihashi et al. (2009)'s algorithm samples directly from the proposal distribution without the accept-reject step, it is not actually sampling from the intended posterior distribution. Because Q approaches the true conditional distribution as the size of the training data increases, however, there may be almost no noticeable difference between using and not using the accept/reject step, though strictly speaking, it is required to guarantee convergence to the the target posterior.

	Unigram		Bigram	
	TF	logProb	TF	logProb
MHS	50.39	**-196.74**	70.93	**-237.24**
PF_1	55.82	-248.21	49.43	-265.40
PF_{16}	62.34	-239.22	50.14	-262.34
PF_{1000}	64.11	-234.87	57.88	-254.17
$PF_{1,100}$	63.17	-245.32	66.88	-257.65
$PF_{16,100}$	68.05	-235.71	70.05	-251.66
$PF_{1,1600}$	**77.06**	-228.79	**74.47**	-249.78

Table 1: Results for both the Unigram and the Bigram model. MHS is a Metropolis-Hastings batch sampler. PF_x is a particle filter with x particles and no rejuvenation. $PF_{x,s}$ is a particle filter with x particles and s rejuvenation steps. TF is token f-score, logProb is the log-probability ($\times 10^3$) of the training-data at the end of learning. Less negative logProb indicates a better solution according to the model, higher TF indicates a better quality segmentation. All results are averaged across 4 runs. Results for the 1000 particle setting are taken from Börschinger and Johnson (2011).

the 16-particle learner with rejuvenation with a 1-particle learner that performs 16 times as many rejuvenation samples. For comparison, we also cite previous results for the 1000-particle learners without rejuvenation reported in Börschinger and Johnson (2011), using their choice of parameters to allow for a direct comparison: $\alpha = 20$ for the Unigram model, $\alpha_0 = 3000$, $\alpha_1 = 100$ for the Bigram model, and we use their base-distribution which differs from the one described in Goldwater et al. (2009) in that it doesn't assume a uniform distribution over segments in the base-distribution but puts a Dirichlet Prior on it.

We apply each learner to the Bernstein-Ratner corpus (Brent, 1999) that is standardly used in the word segmentation literature, which consists of 9790 unsegmented and phonemically transcribed child-directed speech utterances. We evaluate each algorithm in two ways: inference performance, for which the final log-probability of the training data is the criterion, and segmentation performance, for which we consider token f-score to be the best measure, since it indicates how well the actual word tokens in the data are recovered.Note that these two measures can diverge, as previously documented for the Unigram model (Goldwater, 2006) and, less so, for the Bigram model (Pearl et al., 2010). Table 1 gives the results for our experiments.

For both models, adding rejuvenation always improves performance markedly as compared to the corresponding run without rejuvenation both in terms of log-probability and segmentation f-score. Note in particular that for the Bigram model, using 16 particles with 100 rejuvenation steps leads to an improvement in token f-score of more than 10% points over 1000 particles without rejuvenation.

Comparing the 1-particle learner with 1600 rejuvenation steps to the 16-particle learner with 100 rejuvenation steps, for both models the former outperforms the latter in both log-probability and token f-score. This suggests that if one has to trade-off particle number against rejuvenation steps, one may be better off favouring the latter.

Despite the dramatic improvement over not using rejuvenation, there is still a considerable gap between all the incremental learners and the batch sampling algorithm in terms of log-probability. A similar observation was made by Johnson and Goldwater (2009) for incremental initialisation in word segmentation using adaptor grammars. Their *batch* sampler converged on higher token f-score but lower probability solutions in some settings when initialized in an incremental fashion as opposed to randomly. We agree with their suggestion that this may be due to the "greedy" character of an incremental learner.

6 Conclusion and outlook

We have shown that adding rejuvenation to a particle filter improves segmentation scores and log-probabilities. Yet, our incremental algorithm still finds lower probability but high quality token f-scores compared to its batch counterpart. While in principle, increasing the number of rejuvenation steps and particles will make this gap smaller and smaller, we believe the existence of the gap to be interesting in its own right, suggesting a general difference in learning behaviour between batch and incremental learners, especially given the similar results in Johnson and Goldwater (2009). Further research into incremental learning algorithms may help us better understand how processing limitations can affect learning and why this may be beneficial for language acquisition, as suggested, for example, in Newport (1988).

References

Benjamin Börschinger and Mark Johnson. 2011. A particle filter algorithm for bayesian wordsegmentation. In *Proceedings of the Australasian Language Technology Association Workshop 2011*, pages 10–18, Canberra, Australia, December.

Michael R. Brent. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34(1-3):71–105.

Kevin R. Canini, Lei Shi, and Thomas L. Griffiths. 2009. Online inference of topics with latent Dirichlet allocation. In David van Dyk and Max Welling, editors, *Proceeings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 65–72.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2009. A bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21–54.

Sharon Goldwater. 2006. *Nonparametric Bayesian Models of Lexical Acquisition*. Ph.D. thesis, Brown University.

Mark Johnson and Sharon Goldwater. 2009. Improving nonparametric 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*, Boulder, Colorado.

Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. Bayesian inference for pcfgs via markov chain monte carlo. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*.

Daichi Mochihashi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested pitman-yor language modeling. 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 100–108, Suntec, Singapore, August. Association for Computational Linguistics.

Elissa L Newport. 1988. Constraints on learning and their role in language acquisition: Studies of the acquisition of american sign language. *Language Sciences*, 10:147–172.

Lisa Pearl, Sharon Goldwater, and Mark Steyvers. 2010. Online learning mechanisms for bayesian models of word segmentation. *Research on Language and Computation*, 8(2):107–132.

Baselines and Bigrams: Simple, Good Sentiment and Topic Classification

Sida Wang and **Christopher D. Manning**
Department of Computer Science
Stanford University
Stanford, CA 94305
{sidaw,manning}@stanford.edu

Abstract

Variants of Naive Bayes (NB) and Support Vector Machines (SVM) are often used as baseline methods for text classification, but their performance varies greatly depending on the model variant, features used and task/ dataset. We show that: (i) the inclusion of word bigram features gives consistent gains on sentiment analysis tasks; (ii) for short snippet sentiment tasks, NB actually does better than SVMs (while for longer documents the opposite result holds); (iii) a simple but novel SVM variant using NB log-count ratios as feature values consistently performs well across tasks and datasets. Based on these observations, we identify simple NB and SVM variants which outperform most published results on sentiment analysis datasets, sometimes providing a new state-of-the-art performance level.

1 Introduction

Naive Bayes (NB) and Support Vector Machine (SVM) models are often used as baselines for other methods in text categorization and sentiment analysis research. However, their performance varies significantly depending on which variant, features and datasets are used. We show that researchers have not paid sufficient attention to these model selection issues. Indeed, we show that the better variants often outperform recently published state-of-the-art methods on many datasets. We attempt to categorize which method, which variants and which features perform better under which circumstances.

First, we make an important distinction between sentiment classification and topical text classification. We show that the usefulness of bigram features in bag of features sentiment classification has been underappreciated, perhaps because their usefulness is more of a mixed bag for topical text classification tasks. We then distinguish between short snippet sentiment tasks and longer reviews, showing that for the former, NB outperforms SVMs. Contrary to claims in the literature, we show that bag of features models are still strong performers on snippet sentiment classification tasks, with NB models generally outperforming the sophisticated, structure-sensitive models explored in recent work. Furthermore, by combining generative and discriminative classifiers, we present a simple model variant where an SVM is built over NB log-count ratios as feature values, and show that it is a strong and robust performer over all the presented tasks. Finally, we confirm the well-known result that MNB is normally better and more stable than multivariate Bernoulli NB, and the increasingly known result that binarized MNB is better than standard MNB. The code and datasets to reproduce the results in this paper are publicly available. [1]

2 The Methods

We formulate our main model variants as linear classifiers, where the prediction for test case k is

$$y^{(k)} = \text{sign}(\mathbf{w}^T \mathbf{x}^{(k)} + b) \quad (1)$$

Details of the equivalent probabilistic formulations are presented in (McCallum and Nigam, 1998).

Let $\mathbf{f}^{(i)} \in \mathbb{R}^{|V|}$ be the feature count vector for training case i with label $y^{(i)} \in \{-1, 1\}$. V is the

[1] http://www.stanford.edu/~sidaw

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 90–94,
Jeju, Republic of Korea, 8-14 July 2012.

set of features, and $\mathbf{f}_j^{(i)}$ represents the number of occurrences of feature V_j in training case i. Define the count vectors as $\mathbf{p} = \alpha + \sum_{i:y^{(i)}=1} \mathbf{f}^{(i)}$ and $\mathbf{q} = \alpha + \sum_{i:y^{(i)}=-1} \mathbf{f}^{(i)}$ for smoothing parameter α. The log-count ratio is:

$$\mathbf{r} = \log\left(\frac{\mathbf{p}/||\mathbf{p}||_1}{\mathbf{q}/||\mathbf{q}||_1}\right) \quad (2)$$

2.1 Multinomial Naive Bayes (MNB)

In MNB, $\mathbf{x}^{(k)} = \mathbf{f}^{(k)}$, $\mathbf{w} = \mathbf{r}$ and $b = \log(N_+/N_-)$. N_+, N_- are the number of positive and negative training cases. However, as in (Metsis et al., 2006), we find that binarizing $\mathbf{f}^{(k)}$ is better. We take $\mathbf{x}^{(k)} = \hat{\mathbf{f}}^{(k)} = \mathbf{1}\{\mathbf{f}^{(k)} > 0\}$, where $\mathbf{1}$ is the indicator function. $\hat{\mathbf{p}}, \hat{\mathbf{q}}, \hat{\mathbf{r}}$ are calculated using $\hat{\mathbf{f}}^{(i)}$ instead of $\mathbf{f}^{(i)}$ in (2).

2.2 Support Vector Machine (SVM)

For the SVM, $\mathbf{x}^{(k)} = \hat{\mathbf{f}}^{(k)}$, and $\mathbf{w}, b$ are obtained by minimizing

$$\mathbf{w}^T\mathbf{w} + C\sum_i \max(0, 1 - y^{(i)}(\mathbf{w}^T\hat{\mathbf{f}}^{(i)} + b))^2 \quad (3)$$

We find this L2-regularized L2-loss SVM to work the best and L1-loss SVM to be less stable. The LIBLINEAR library (Fan et al., 2008) is used here.

2.3 SVM with NB features (NBSVM)

Otherwise identical to the SVM, except we use $\mathbf{x}^{(k)} = \tilde{\mathbf{f}}^{(k)}$, where $\tilde{\mathbf{f}}^{(k)} = \hat{\mathbf{r}} \circ \hat{\mathbf{f}}^{(k)}$ is the elementwise product. While this does very well for long documents, we find that an interpolation between MNB and SVM performs excellently for all documents and we report results using this model:

$$\mathbf{w}' = (1 - \beta)\bar{w} + \beta\mathbf{w} \quad (4)$$

where $\bar{w} = ||\mathbf{w}||_1/|V|$ is the mean magnitude of $\mathbf{w}$, and $\beta \in [0, 1]$ is the interpolation parameter. This interpolation can be seen as a form of regularization: trust NB unless the SVM is very confident.

3 Datasets and Task

We compare with published results on the following datasets. Detailed statistics are shown in table 1.

RT-s: Short movie reviews dataset containing one sentence per review (Pang and Lee, 2005).

Dataset	(N_+, N_-)	l	CV	$\lvert V\rvert$	Δ
RT-s	(5331,5331)	21	10	21K	0.8
CR	(2406,1366)	20	10	5713	1.3
MPQA	(3316,7308)	3	10	6299	0.8
Subj.	(5000,5000)	24	10	24K	0.8
RT-2k	(1000,1000)	787	10	51K	1.5
IMDB	(25k,25k)	231	N	392K	0.4
AthR	(799,628)	345	N	22K	2.9
XGraph	(980,973)	261	N	32K	1.8
BbCrypt	(992,995)	269	N	25K	0.5

Table 1: Dataset statistics. (N_+, N_-): number of positive and negative examples. l: average number of words per example. CV: number of cross-validation splits, or N for train/test split. $|V|$: the vocabulary size. Δ: upper-bounds of the differences required to be statistically significant at the $p < 0.05$ level.

CR: Customer review dataset (Hu and Liu, 2004) processed like in (Nakagawa et al., 2010).[2]

MPQA: Opinion polarity subtask of the MPQA dataset (Wiebe et al., 2005).[3]

Subj: The subjectivity dataset with subjective reviews and objective plot summaries (Pang and Lee, 2004).

RT-2k: The standard 2000 full-length movie review dataset (Pang and Lee, 2004).

IMDB: A large movie review dataset with 50k full-length reviews (Maas et al., 2011).[4]

AthR, XGraph, BbCrypt: Classify pairs of newsgroups in the 20-newsgroups dataset with all headers stripped off (the third (18828) version[5]), namely: alt.atheism vs. religion.misc, comp.windows.x vs. comp.graphics, and rec.sport.baseball vs. sci.crypt, respectively.

4 Experiments and Results

4.1 Experimental setup

We use the provided tokenizations when they exist. If not, we split at spaces for unigrams, and we filter out anything that is not [A-Za-z] for bigrams. We do

[2] http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
[3] http://www.cs.pitt.edu/mpqa/
[4] http://ai.stanford.edu/~amaas/data/sentiment
[5] http://people.csail.mit.edu/jrennie/20Newsgroups

not use stopwords, lexicons or other resources. All results reported use $\alpha = 1$, $C = 1, \beta = 0.25$ for NBSVM, and $C = 0.1$ for SVM.

For comparison with other published results, we use either 10-fold cross-validation or train/test split depending on what is standard for the dataset. The CV column of table 1 specifies what is used. The standard splits are used when they are available. The approximate upper-bounds on the difference required to be statistically significant at the $p < 0.05$ level are listed in table 1, column Δ.

4.2 MNB is better at snippets

(Moilanen and Pulman, 2007) suggests that while "statistical methods" work well for datasets with hundreds of words in each example, they cannot handle snippets datasets and some rule-based system is necessary. Supporting this claim are examples such as *not an inhumane monster*[6], or *killing cancer* that express an overall positive sentiment with negative words.

Some previous work on classifying snippets include using pre-defined polarity reversing rules (Moilanen and Pulman, 2007), and learning complex models on parse trees such as in (Nakagawa et al., 2010) and (Socher et al., 2011). These works seem promising as they perform better than many sophisticated, rule-based methods used as baselines in (Nakagawa et al., 2010). However, we find that several NB/SVM variants in fact do better than these state-of-the-art methods, even compared to methods that use lexicons, reversal rules, or unsupervised pretraining. The results are in table 2.

Our SVM-uni results are consistent with BoF-noDic and BoF-w/Rev used in (Nakagawa et al., 2010) and BoWSVM in (Pang and Lee, 2004). (Nakagawa et al., 2010) used a SVM with second-order polynomial kernel and additional features. With the only exception being MPQA, MNB performed better than SVM in all cases.[7]

Table 2 show that a linear SVM is a weak baseline for snippets. MNB (and NBSVM) are much better on sentiment snippet tasks, and usually better than other published results. Thus, we find the hypothesis that rule-based systems have an edge for snippet datasets to be false. MNB is stronger for snippets than for longer documents. While (Ng and Jordan, 2002) showed that NB is better than SVM/logistic regression (LR) with few training cases, we show that MNB is also better with short documents. In contrast to their result that an SVM usually beats NB when it has more than 30–50 training cases, we show that MNB is still better on snippets even with relatively large training sets (9k cases).

Method	RT-s	MPQA	CR	Subj.
MNB-uni	77.9	85.3	79.8	**92.6**
MNB-bi	**79.0**	**86.3**	80.0	**<u>93.6</u>**
SVM-uni	76.2	86.1	79.0	90.8
SVM-bi	77.7	**<u>86.7</u>**	80.8	91.7
NBSVM-uni	**78.1**	85.3	80.5	92.4
NBSVM-bi	**<u>79.4</u>**	**86.3**	**<u>81.8</u>**	**93.2**
RAE	76.8	85.7	–	–
RAE-pretrain	77.7	**86.4**	–	–
Voting-w/Rev.	63.1	81.7	74.2	–
Rule	62.9	81.8	74.3	–
BoF-noDic.	75.7	81.8	79.3	–
BoF-w/Rev.	76.4	84.1	**81.4**	–
Tree-CRF	77.3	86.1	**81.4**	–
BoWSVM	–	–	–	90.0

Table 2: Results for snippets datasets. Tree-CRF: (Nakagawa et al., 2010) RAE: Recursive Autoencoders (Socher et al., 2011). RAE-pretrain: train on Wikipedia (Collobert and Weston, 2008). "Voting" and "Rule": use a sentiment lexicon and hard-coded reversal rules. "w/Rev": "the polarities of phrases which have odd numbers of reversal phrases in their ancestors". The top 3 methods are in **bold** and the best is also **<u>underlined</u>**.

4.3 SVM is better at full-length reviews

As seen in table 1, the RT-2k and IMDB datasets contain much longer reviews. Compared to the excellent performance of MNB on snippet datasets, the many poor assumptions of MNB pointed out in (Rennie et al., 2003) become more crippling for these longer documents. SVM is much stronger than MNB for the 2 full-length sentiment analysis tasks, but still worse than some other published results. However, NBSVM either exceeds or approaches previous state-of-the art methods, even the

[6] A positive example from the RT-s dataset.

[7] We are unsure, but feel that MPQA may be less discriminative, since the documents are extremely short and all methods perform similarly.

Our results	RT-2k	IMDB	Subj.
MNB-uni	83.45	83.55	**92.58**
MNB-bi	85.85	86.59	**93.56**
SVM-uni	86.25	86.95	90.84
SVM-bi	87.40	**89.16**	91.74
NBSVM-uni	87.80	88.29	92.40
NBSVM-bi	**89.45**	**91.22**	**93.18**
BoW (bnc)	85.45	87.8	87.77
BoW (bΔt$'$c)	85.8	88.23	85.65
LDA	66.7	67.42	66.65
Full+BoW	87.85	88.33	88.45
Full+Unlab'd+BoW	**88.9**	88.89	88.13
BoWSVM	87.15	–	90.00
Valence Shifter	86.2	–	–
tf.Δidf	88.1	–	–
Appr. Taxonomy	**90.20**	–	–
WRRBM	–	87.42	–
WRRBM + BoW(bnc)	–	**89.23**	–

Table 3: Results for long reviews (RT-2k and IMDB). The snippet dataset Subj. is also included for comparison. Results in rows 7-11 are from (Maas et al., 2011). **BoW**: linear SVM on bag of words features. **bnc**: **b**inary, **n**o idf, **c**osine normalization. Δ**t**$'$: smoothed delta idf. **Full**: the full model. **Unlab'd**: additional unlabeled data. **BoWSVM**: bag of words SVM used in (Pang and Lee, 2004). **Valence Shifter**: (Kennedy and Inkpen, 2006). **tf.Δidf**: (Martineau and Finin, 2009). **Appraisal Taxonomy**: (Whitelaw et al., 2005). **WRRBM**: Word Representation Restricted Boltzmann Machine (Dahl et al., 2012).

ones that use additional data. These sentiment analysis results are shown in table 3.

4.4 Benefits of bigrams depends on the task

Word bigram features are not that commonly used in text classification tasks (hence, the usual term, "bag of words"), probably due to their having mixed and overall limited utility in topical text classification tasks, as seen in table 4. This likely reflects that certain topic keywords are indicative alone. However, in both tables 2 and 3, adding bigrams *always* improved the performance, and often gives better results than previously published.[8] This presumably reflects that in sentiment classification there are much bigger gains from bigrams, because they can capture modified verbs and nouns.

[8]However, adding **trigrams** hurts slightly.

Method	AthR		XGraph		BbCrypt	
MNB-uni	85.0		90.0		**99.3**	
MNB-bi	**85.1**	+0.1	**91.2**	+1.2	99.4	+0.1
SVM-uni	82.6		85.1		98.3	
SVM-bi	83.7	+1.1	86.2	+0.9	97.7	−0.5
NBSVM-uni	**87.9**		**91.2**		99.7	
NBSVM-bi	**87.7**	−0.2	**90.7**	−0.5	99.5	−0.2
ActiveSVM	–		90		99	
DiscLDA	83		–		–	

Table 4: On 3 20-newsgroup subtasks, we compare to DiscLDA (Lacoste-Julien et al., 2008) and ActiveSVM (Schohn and Cohn, 2000).

4.5 NBSVM is a robust performer

NBSVM performs well on snippets and longer documents, for sentiment, topic and subjectivity classification, and is often better than previously published results. Therefore, NBSVM seems to be an appropriate and very strong baseline for sophisticated methods aiming to beat a bag of features.

One disadvantage of NBSVM is having the interpolation parameter β. The performance on longer documents is virtually identical (within 0.1%) for $\beta \in [¼, 1]$, while $\beta = ¼$ is on average 0.5% better for snippets than $\beta = 1$. Using $\beta \in [¼, ½]$ makes the NBSVM more robust than more extreme values.

4.6 Other results

Multivariate Bernoulli NB (BNB) usually performs worse than MNB. The only place where BNB is comparable to MNB is for snippet tasks using only unigrams. In general, BNB is less stable than MNB and performs up to 10% worse. Therefore, benchmarking against BNB is untrustworthy, cf. (McCallum and Nigam, 1998).

For MNB and NBSVM, using the binarized MNB $\hat{\mathbf{f}}$ is slightly better (by 1%) than using the raw count feature $\mathbf{f}$. The difference is negligible for snippets.

Using logistic regression in place of SVM gives similar results, and some of our results can be viewed more generally in terms of generative vs. discriminative learning.

References

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*.

George E. Dahl, Ryan P. Adams, and Hugo Larochelle. 2012. Training restricted boltzmann machines on word observations. *arXiv:1202.5695v1 [cs.LG]*.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, June.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings ACM SIGKDD*, pages 168–177.

Alistair Kennedy and Diana Inkpen. 2006. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22.

Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. 2008. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *Proceedings of NIPS*, pages 897–904.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of ACL*.

Justin Martineau and Tim Finin. 2009. Delta tfidf: An improved feature space for sentiment analysis. In *Proceedings of ICWSM*.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop*, pages 41–48.

Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. 2006. Spam filtering with naive bayes - which naive bayes? In *Proceedings of CEAS*.

Karo Moilanen and Stephen Pulman. 2007. Sentiment composition. In *Proceedings of RANLP*, pages 378–382, September 27-29.

Tetsuji Nakagawa, Kentaro Inui, and Sadao Kurohashi. 2010. Dependency tree-based sentiment classification using CRFs with hidden variables. In *Proceedings of ACL:HLT*.

Andrew Y Ng and Michael I Jordan. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Proceedings of NIPS*, volume 2, pages 841–848.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL*.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*.

Jason D. Rennie, Lawrence Shih, Jaime Teevan, and David R. Karger. 2003. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of ICML*, pages 616–623.

Greg Schohn and David Cohn. 2000. Less is more: Active learning with support vector machines. In *Proceedings of ICML*, pages 839–846.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proceedings of EMNLP*.

Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal taxonomies for sentiment analysis. In *Proceedings of CIKM-05*.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Automatically Learning Measures of Child Language Development

Sam Sahakian
University of Wisconsin - Madison
sahakian@cs.wisc.edu

Benjamin Snyder
University of Wisconsin - Madison
bsnyder@cs.wisc.edu

Abstract

We propose a new approach for the creation of child language development metrics. A set of linguistic features is computed on child speech samples and used as input in two age prediction experiments. In the first experiment, we learn a child-specific metric and predicts the ages at which speech samples were produced. We then learn a more general developmental index by applying our method across children, predicting relative temporal orderings of speech samples. In both cases we compare our results with established measures of language development, showing improvements in age prediction performance.

1 Introduction

The rapid childhood development from a seemingly blank slate to language mastery is a puzzle that linguists and psychologists continue to ponder. While the precise mechanism of language learning remains poorly understood, researchers have developed measures of developmental language progress using child speech patterns. These metrics provide a means of diagnosing early language disorders. Besides this practical benefit, precisely measuring grammatical development is a step towards understanding the underlying language learning process.

Previous NLP work has sought to automate the calculation of handcrafted developmental metrics proposed by psychologists and linguists. In this paper, we investigate a more fundamental question: Can we use machine learning techniques to create a more robust developmental measure itself? If so, how well would such a measure generalize across children? This last question touches on an underlying assumption made in much of the child language literature– that while children progress grammatically at different rates, they follow fixed stages in their development. If a developmental index automatically learned from one set of children could be accurately applied to others, it would vindicate this assumption of shared developmental paths.

Several metrics of language development have been set forth in the psycholinguistics literature. Standard measures include Mean Length of Utterance (MLU) (Brown, 1973)– the average length in morphemes of conversational turns, Index of Productive Syntax (IPSYN) (Scarborough, 1990)– a multi-tiered scoring process where over 60 individual features are counted by hand and combined into tiered scores, and D-Level (Rosenberg et al., 1987; Covington et al., 2006)– a score for individual sentences based on the observed presence of key syntactic structures. Today, these hand-crafted metrics persist as measurements of child language development, each taking a slightly different angle to assess the same question: Exactly how much grammatical knowledge does a young learner possess?

NLP technology has been applied to help automate the otherwise tedious calculation of these measures. Computerized Profiling (CP) (Long and Channell, 2001) is a software package that produces semi-automated language assessments, using part-of-speech tagging and human supervision. In response to its limited depth of analysis and the necessity for human supervision in CP, there have since

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 95–99,
Jeju, Republic of Korea, 8-14 July 2012.

	D-Level	Article Count	"Be" Count	Fn. / Content	Prep. Count	Word Freq.	Depth	MLU
Adam	0.798	0.532	0.817	0.302	0.399	0.371	0.847	0.855
Abe	0.633	0.479	0.591	0.144	0.269	0.413	0.534	0.625
Ross	0.252	0.153	-0.061	0.125	0.314	0.209	0.134	0.165
Peter	0.371	0.429	0.781	0.562	0.638	0.657	0.524	0.638
Naomi	0.812	0.746	0.540	0.652	0.504	0.609	0.710	0.710
Sarah	0.829	0.550	0.733	0.382	0.654	0.570	0.731	0.808
Nina	0.824	0.758	0.780	0.560	0.451	0.429	0.780	0.890
Mean:	0.646	0.521	0.597	0.390	0.461	0.465	0.609	0.670

Table 1: τ of each feature versus time, for each individual child. In this and all following tables, traditional developmental metrics are shaded.

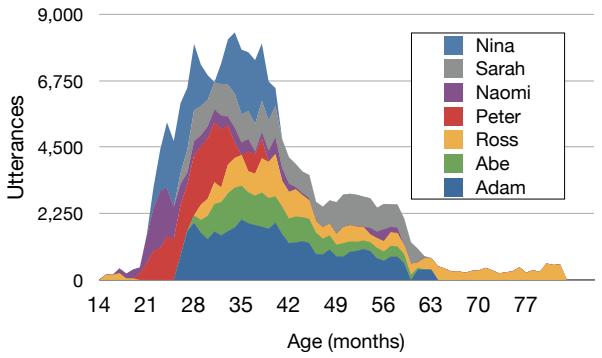


Figure 1: Number of utterances across ages of each child in our corpus. Sources: Nina (Suppes, 1974), Sarah (Brown, 1973), Naomi (Sachs, 1983), Peter (Bloom et al., 1974; Bloom et al., 1975), Ross (MacWhinney, 2000), Abe (Kuczaj, 1977) and Adam (Brown, 1973)

been implementations of completely automated assessments of IPSYN (Sagae et al., 2005) and D-Level (Lu, 2009) which take advantage of automatic parsing and achieve results comparable to manual assessments. Likewise, in the ESL domain, Chen and Zechner (2011) automate the evaluation of syntactic complexity of non-native speech.

Thus, it has been demonstrated that NLP techniques can compute existing scores of language proficiency. However, the definition of first-language developmental metrics has as yet been left up to human reasoning. In this paper, we consider the automatic induction of more accurate developmental metrics using child language data. We extract features from longitudinal child language data and conduct two sets of experiments. For individual children, we use least-squares regression over our features to predict the age of a held-out language sample. We find that on average, existing single metrics of development are outperformed by a weighted combination of our features.

In our second set of experiments, we investigate whether metrics can be learned across children. To do so, we consider a speech sample ordering task. We use optimization techniques to learn weightings over features that allow generalization across children. Although traditional measures like MLU and D-level perform well on this task, we find that a learned combination of features outperforms any single pre-defined developmental score.

2 Data

To identify trends in child language learning we need a corpus of child speech samples, which we take from the CHILDES database (MacWhinney, 2000). CHILDES is a collection of corpora from many studies of child language based on episodic speech data. Since we are interested in development over time, our corpus consists of seven longitudinal studies of individual children. Data for each child is grouped and sorted by the child's age in months, so that we have a single data point for each month in which a child was observed. The size of our data set, broken down by child, is shown in Figure 1.

We take advantage of automatic dependency parses bundled with the CHILDES transcripts (Sagae et al., 2007) and harvest features that should be informative and complementary in assessing grammatical knowledge. We first note three standard measures of language development: *(i)* MLU, a measure of utterance length, *(ii)* mean depth of dependency parse trees, a measure of syntactic complexity similar to that of Yngve (1960), and *(iii)* D-level, a measure of linguistic competence based on observations of syntactic constructions.

Beyond the three traditional developmental metrics, we record five additional features. We count two of Brown's (1973) obligatory morphemes — articles and contracted auxiliary "be" verbs — as well as occurrences of any preposition. These counted features are normalized by a child's total number of utterances at a given age. Finally, we include two vocabulary-centric features: Average word fre-

	D-Level	Depth	MLU	All Features
Adam	14.037	14.149	**11.128**	14.175
Abe	34.69	44.701	**34.509**	39.931
Ross	329.64	336.612	345.046	**244.071**
Peter	23.58	13.045	**8.245**	24.128
Naomi	**24.458**	28.426	34.956	45.036
Sarah	12.503	20.878	13.905	**6.989**
Nina	7.654	6.477	4.255	**3.96**
Mean	63.795	66.327	64.578	**54.041**

Table 2: Mean squared error from 10-fold cross validation of linear regression on individual children. The lowest error for each child is shown in bold.

quency (i.e. how often a word is used in a standard corpus) as indicated by CELEX (Baayen et al., 1995), and the child's ratio of function words (determiners, pronouns, prepositions, auxiliaries and conjunctions) to content words.

To validate a developmental measure, we rely on the assumption that a perfect metric should increase monotonically over time. We therefore calculate Kendall's Tau coefficient (τ) between an ordering of each child's speech samples by age, and an ordering by the given scoring metric. The τ coefficient is a measure of rank correlation where two identical orderings receive a τ of 1, complete opposite orderings receive a τ of -1, and independent orderings are expected to receive a τ of zero. The τ coefficients for each of our 8 features individually applied to the 7 children are shown in Table 1.

We note that the pre-defined indices of language development — MLU, tree depth and D-Level — perform the ordering task most accurately. To illustrate the degree of variance between children and features, we also include plots of each child's D-Level and contracted auxiliary "be" usage in Figure 2.

3 Experiments

Learning Individual Child Metrics Our first task is to predict the age at which a held-out speech sample was produced, given a set of age-stamped samples from the same child. We perform a least squares regression on each child, treating age as the dependent variable, and our features as independent variables. Each data set is split into 10 random folds of 90% training and 10% test data. Mean squared error is reported in Table 2. On average, our regression achieves lower error than any individual feature by itself.

MLU	All Features	MLU & Fn. / Content
0.7456	0.7457	**0.7780**

Table 3: Average τ of orderings produced by MLU (the best traditional index) and our learned metric, versus true chronological order. Highest τ is shown in bold.

Learning General Metrics Across Children To produce a universal metric of language development like MLU or D-Level, we train on data pooled across many children. For each of 7 folds, a single child's data is separated as a test set while the remaining children are used for training. Since Ross is the only child with samples beyond 62 months, we do not attempt to learn a general measure of language development at these ages, but rather remove these data points.

Unlike the individual-child case, we do not predict absolute ages based on speech samples, as each child is expected to learn at a different rate. Instead, we learn an ordering model which attempts to place each sample in its relative place in time. The model computes a score from a weighted quadratic combination of our features and orders the samples based on their computed scores. To learn the parameters of the model, we seek to maximize the Kendall τ between true and predicted orderings, summed over the training children. We pass this objective function to Nelder-Mead (Nelder and Mead, 1965), a standard gradient-free optimization algorithm. Nelder-Mead constructs a simplex at its initial guess of parameter values and iteratively makes small shifts in the simplex to satisfy a descent condition until a local maximum is reached.

We report the average Kendall τ achieved by this algorithm over several feature combinations in Table 3. Because we modify our data set in this experiment, for comparison we also show the average Kendall τ achieved by MLU on the truncated data.

4 Discussion

Our first set of experiments verified that we can achieve a decrease in mean squared error over existing metrics in a child-specific age prediction task. However, the results of this experiment are skewed

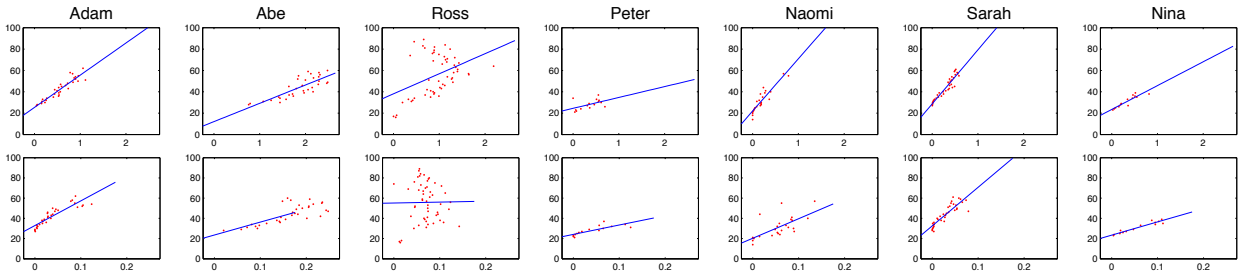


Figure 2: Child age plotted against D-Level (top) and counts of contracted auxiliary "be" (bottom) with best fit lines. Since our regression predicts child age, age in months is plotted on the y-axis.

in favor of the learned metric by the apparent difficulty of predicting Ross's age. As demonstrated in Figure 2, Ross's data exhibits major variance, and also includes data from later ages than that of the other children. It is well known that MLU's performance as a measure of linguistic ability quickly drops off with age.

During our first experiment, we also attempted to capture more nuanced learning curves than the linear case. Specifically, we anticipated that learning over time should follow an S-shaped curve. This follows from observations of a "fast mapping" spurt in child word learning (Woodward et al., 1994), and the idea that learning must eventually level off as mastery is attained. To allow our model to capture non-linear learning rates, we fit logit and quadratic functions to the data. Despite the increased freedom, only Nina's predictions benefited from these more complex models. With every other child, these functions fit the data to a linear section of the curve and yielded much larger errors than simple linear regression. The preference towards linearity may be due to the limited time span of our data. With higher ages, the leveling off of linguistic performance would need to be modeled.

In our second set of experiments, we attempted to learn a general metric across children. Here we also achieved positive results with simple methods, just edging out established measures of language development. The generality of our learned metric supports the hypothesis that children follow similar paths of language development. Although our learned solution is slightly more favorable than preexisting metrics, it performs very little learning. Using all features, learned parameter weights remain at or extremely close to the starting point of 1.

Through trial and error, we discovered we could improve performance by omitting certain features. In Table 3, we report the best discovered feature combination including only two relatively uncorrelated features, MLU and function/content word ratio. If downweighting some features yields a better result, we would expect to discover that with our optimization algorithm, but this evidently not the case, perhaps due to our limited sample of 7 children.

The fact that weights move so little suggests that our best result is stuck in a local maximum. To investigate this, we also experimented with Differential Evolution (Storn and Price, 1997) and SVM-ranking (Joachims, 2002), the former a global optimization technique, and the latter a method developed specifically to learn orderings. Although these algorithms are more willing to adjust parameter weights and theoretically should not get stuck in local maxima, they are still edged out in performance by Nelder-Mead. It may be that the early stopping of Nelder-Mead serves as a sort of smoothing in this very small data-set of 7 children.

Our improvements over hand-crafted measures of language development show promise. In the case of individual children, we outperform existing measures of development, especially past the early stages of development when MLU ceases to correlate with age. Our attempts to learn a metric across children met with more limited success. However, when we restricted our regression to two of the least correlated features, MLU and the function/content word ratio, we were able to beat manually created metrics. These results suggest that more sophisticated models and techniques combined with more data could lead to more accurate metrics as well as insights into the language learning process.

References

R.H. Baayen, R. Piepenbrock, and L. Gulikers. 1995. The CELEX lexical database (release 2)[cd-rom]. *Philadelphia, PA: Linguistic Data Consortium, University of Pennsylvania [Distributor]*.

L. Bloom, L. Hood, and P. Lightbown. 1974. Imitation in language development: If, when, and why. *Cognitive Psychology*, 6(3):380–420.

L. Bloom, P. Lightbown, L. Hood, M. Bowerman, M. Maratsos, and M.P. Maratsos. 1975. Structure and variation in child language. *Monographs of the Society for Research in Child Development*, pages 1–97.

R. Brown. 1973. *A First Language: The Early Stages.* Harvard U. Press.

M. Chen and K. Zechner. 2011. Computing and evaluating syntactic complexity features for automated scoring of spontaneous non-native speech. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 722–731.

M.A. Covington, C. He, C. Brown, L. Naci, and J. Brown. 2006. How complex is that sentence? a proposed revision of the Rosenberg and Abbeduto D-level scale. *Research Report, AI Center, University of Georgia*.

T. Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142. ACM.

S.A. Kuczaj. 1977. The acquisition of regular and irregular past tense forms. *Journal of Verbal Learning and Verbal Behavior*, 16(5):589–600.

S.H. Long and R.W. Channell. 2001. Accuracy of four language analysis procedures performed automatically. *American Journal of Speech-Language Pathology*, 10(2):180.

X. Lu. 2009. Automatic measurement of syntactic complexity in child language acquisition. *International Journal of Corpus Linguistics*, 14(1):3–28.

B. MacWhinney. 2000. *The CHILDES project: Tools for analyzing talk*, volume 2. Psychology Press.

J.A. Nelder and R. Mead. 1965. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313.

S. Rosenberg, L. Abbeduto, et al. 1987. Indicators of linguistic competence in the peer group conversational behavior of mildly retarded adults. *Applied Psycholinguistics*, 8(1):19–32.

J. Sachs. 1983. Talking about the there and then: The emergence of displaced reference in parent-child discourse. *Childrens Language*, 4.

K. Sagae, A. Lavie, and B. MacWhinney. 2005. Automatic measurement of syntactic development in child language. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 197–204. Association for Computational Linguistics.

K. Sagae, E. Davis, A. Lavie, B. MacWhinney, and S. Wintner. 2007. High-accuracy annotation and parsing of CHILDES transcripts. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*, pages 25–32. Association for Computational Linguistics.

H.S. Scarborough. 1990. Index of productive syntax. *Applied Psycholinguistics*, 11(1):1–22.

R. Storn and K. Price. 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359.

P. Suppes. 1974. The semantics of children's language. *American Psychologist*, 29(2):103.

A.L. Woodward, E.M. Markman, and C.M. Fitzsimmons. 1994. Rapid word learning in 13-and 18-month-olds. *Developmental Psychology*, 30(4):553.

V.H. Yngve. 1960. A model and an hypothesis for language structure. *Proceedings of the American Philosophical Society*, 104(5):444–466.

A Comparative Study of Target Dependency Structures for Statistical Machine Translation

Xianchao Wu[*], Katsuhito Sudoh, Kevin Duh[†], Hajime Tsukada, Masaaki Nagata
NTT Communication Science Laboratories, NTT Corporation
2-4 Hikaridai Seika-cho, Soraku-gun Kyoto 619-0237 Japan
wuxianchao@gmail.com,sudoh.katsuhito@lab.ntt.co.jp,
kevinduh@is.naist.jp,{tsukada.hajime,nagata.masaaki}@lab.ntt.co.jp

Abstract

This paper presents a comparative study of target dependency structures yielded by several state-of-the-art linguistic parsers. Our approach is to measure the impact of these non-isomorphic dependency structures to be used for string-to-dependency translation. Besides using traditional dependency parsers, we also use the dependency structures transformed from PCFG trees and predicate-argument structures (PASs) which are generated by an HPSG parser and a CCG parser. The experiments on Chinese-to-English translation show that the HPSG parser's PASs achieved the best dependency and translation accuracies.

1 Introduction

Target language side dependency structures have been successfully used in statistical machine translation (SMT) by Shen et al. (2008) and achieved state-of-the-art results as reported in the NIST 2008 Open MT Evaluation workshop and the NTCIR-9 Chinese-to-English patent translation task (Goto et al., 2011; Ma and Matsoukas, 2011). A primary advantage of dependency representations is that they have a natural mechanism for representing discontinuous constructions, which arise due to long-distance dependencies or in languages where grammatical relations are often signaled by morphology instead of word order (McDonald and Nivre, 2011).

It is known that dependency-style structures can be transformed from a number of linguistic structures. For example, using the constituent-to-dependency conversion approach proposed by Johansson and Nugues (2007), we can easily yield dependency trees from PCFG style trees. A semantic dependency representation of a whole sentence, predicate-argument structures (PASs), are also included in the output trees of (1) a state-of-the-art head-driven phrase structure grammar (HPSG) (Pollard and Sag, 1994; Sag et al., 2003) parser, Enju[1] (Miyao and Tsujii, 2008) and (2) a state-of-the-art CCG parser[2] (Clark and Curran, 2007). The motivation of this paper is to investigate the impact of these non-isomorphic dependency structures to be used for SMT. That is, we would like to provide a comparative evaluation of these dependencies in a string-to-dependency decoder (Shen et al., 2008).

2 Gaining Dependency Structures

2.1 Dependency tree

We follow the definition of *dependency graph* and *dependency tree* as given in (McDonald and Nivre, 2011). A dependency graph G for sentence s is called a *dependency tree* when it satisfies, (1) the nodes cover all the words in s besides the ROOT; (2) one node can have one and only one head (word) with a determined syntactic role; and (3) the ROOT of the graph is reachable from all other nodes.

For extracting string-to-dependency transfer rules, we use *well-formed* dependency structures, either fixed or floating, as defined in (Shen et al., 2008). Similarly, we ignore the syntactic roles

[*] Now at Baidu Inc.
[†] Now at Nara Institute of Science & Technology (NAIST)

[1] http://www-tsujii.is.s.u-tokyo.ac.jp/enju/index.html
[2] http://groups.inf.ed.ac.uk/ccg/software.html

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 100–104,
Jeju, Republic of Korea, 8-14 July 2012.

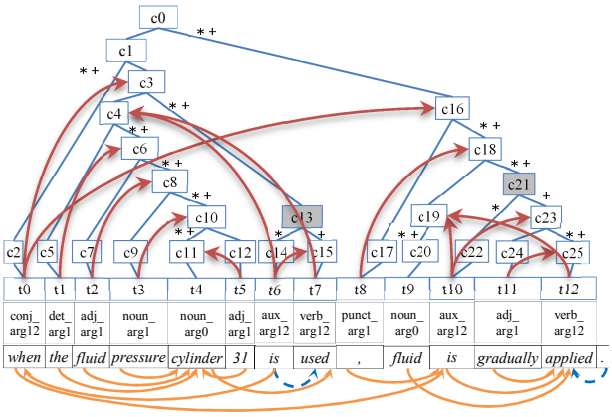


Figure 1: HPSG tree of an example sentence. '*'/'+'=syntactic/semantic heads. Arrows in red (upper)= PASs, orange (bottom)=word-level dependencies generated from PASs, blue=newly appended dependencies.

both during rule extracting and target dependency language model (LM) training.

2.2 Dependency parsing

Graph-based and transition-based are two predominant paradigms for data-driven dependency parsing. The MST parser (McDonald et al., 2005) and the Malt parser (Nivre, 2003) stand for two typical parsers, respectively. Parsing accuracy comparison and error analysis under the CoNLL-X dependency shared task data (Buchholz and Marsi, 2006) have been performed by McDonald and Nivre (2011). Here, we compare them on the SMT tasks through parsing the real-world SMT data.

2.3 PCFG parsing

For PCFG parsing, we select the Berkeley parser (Petrov and Klein, 2007). In order to generate word-level dependency trees from the PCFG tree, we use the LTH constituent-to-dependency conversion tool[3] written by Johansson and Nugues (2007). The head finding rules[4] are according to Magerman (1995) and Collins (1997). Similar approach has been originally used by Shen et al. (2008).

2.4 HPSG parsing

In the Enju English HPSG grammar (Miyao et al., 2003) used in this paper, the semantic content of a sentence/phrase is represented by a PAS. In an HPSG tree, each leaf node generally introduces a predicate, which is represented by the pair made up of the lexical entry feature and predicate type feature. The arguments of a predicate are designated by the arrows from the argument features in a leaf node to non-terminal nodes (e.g., t0→c3, t0→c16).

Since the PASs use the non-terminal nodes in the HPSG tree (Figure 1), this prevents their direct usage in a string-to-dependency decoder. We thus need an algorithm to transform these phrasal predicate-argument dependencies into a word-to-word dependency tree. Our algorithm (refer to Figure 1 for an example) for changing PASs into word-based dependency trees is as follows:

1. *finding*, i.e., find the syntactic/semantic head word of each argument node through a bottom-up traversal of the tree;

2. *mapping*, i.e., determine the *arc directions* (among a predicate word and the syntactic/semantic head words of the argument nodes) for each predicate type according to Table 1. Then, a dependency graph will be generated;

3. *checking*, i.e., post modifying the dependency graph according to the definition of *dependency tree* (Section 2.1).

Table 1 lists the mapping from HPSG's PAS types to word-level dependency arcs. Since a non-terminal node in an HPSG tree has two kinds of heads, syntactic or semantic, we will generate two dependency graphs after mapping. We use "PAS+syn" to represent the dependency trees generated from the HPSG PASs guided by the syntactic heads. For semantic heads, we use "PAS+sem".

For example, refer to t0 = when in Figure 1. Its arg1 = c16 (with syntactic head t10), arg2 = c3 (with syntactic head t6), and PAS type = conj_arg12. In Table 1, this PAS type corresponds to arg2→pred→arg1, then the result word-level dependency is t6(is)→t0(when)→t10(is).

We need to post modify the dependency graph after applying the mapping, since it is not guaranteed to be a dependency tree. Referring to the definition of dependency tree (Section 2.1), we need the strategy for (1) selecting only one head from multiple

[3]http://nlp.cs.lth.se/software/treebank_converter/

[4]http://www.cs.columbia.edu/ mcollins/papers/heads

PAS Type	Dependency Relation
adj_arg1[2]	[arg2 →] pred → arg1
adj_mod_arg1[2]	[arg2 →] pred → arg1 → mod
aux[_mod]_arg12	arg1/pred → arg2 [→ mod]
conj_arg1[2[3]]	[arg2[/arg3]] → pred → arg1
comp_arg1[2]	pred → arg1 [→ arg2]
comp_mod_arg1	arg1 → pred → mod
noun_arg1	pred → arg1
noun_arg[1]2	arg2 → pred [→ arg1]
poss_arg[1]2	pred → arg2 [→ arg1]
prep_arg12[3]	arg2[/arg3] → pred → arg1
prep_mod_arg12[3]	arg2[/arg3] → pred → arg1 → mod
quote_arg[1]2	[arg1 →] pred → arg2
quote_arg[1]23	[arg1/]arg3 → pred → arg2
lparen_arg123	pred/arg2 → arg3 → arg1
relative_arg1[2]	[arg2 →] pred → arg1
verb_arg1[2[3[4]]]	arg1[/arg2[/arg3[/arg4]]] → pred
verb_mod_arg1[2[3[4]]]	arg1[/arg2[/arg3[/arg4]]]→pred→mod
app_arg12,coord_arg12	arg2/pred → arg1
det_arg1,it_arg1,punct_arg1	pred → arg1
dtv_arg2	pred → arg2
lgs_arg2	arg2 → pred

Table 1: Mapping from HPSG's PAS types to dependency relations. Dependent(s) → head(s), / = and, [] = optional.

heads and (2) appending dependency relations for those words/punctuation that do not have any head. When one word has multiple heads, we only keep one. The selection strategy is that, if this arc was deleted, it will cause the biggest number of words that can not reach to the root word anymore. In case of a tie, we greedily pack the arc that connect two words w_i and w_j where $|i-j|$ is the biggest. For all the words and punctuation that do not have a head, we greedily take the root word of the sentence as their heads. In order to fully use the training data, if there are directed cycles in the result dependency graph, we still use the graph in our experiments, where only partial dependency arcs, i.e., those target flat/hierarchical phrases attached with well-formed dependency structures, can be used during translation rule extraction.

2.5 CCG parsing

We also use the predicate-argument dependencies generated by the CCG parser developed by Clark and Curran (2007). The algorithm for generating word-level dependency tree is easier than processing the PASs included in the HPSG trees, since the word level predicate-argument relations have already been included in the output of CCG parser. The mapping from predicate types to the gold-standard grammatical relations can be found in Table 13 in (Clark and Curran, 2007). The post-processing is like that described for HPSG parsing, except we greedily use the MST's sentence root when we can not determine it based on the CCG parser's PASs.

3 Experiments

3.1 Setup

We re-implemented the string-to-dependency decoder described in (Shen et al., 2008). Dependency structures from non-isomorphic syntactic/semantic parsers are separately used to train the transfer rules as well as target dependency LMs. For intuitive comparison, an outside SMT system is Moses (Koehn et al., 2007).

For Chinese-to-English translation, we use the parallel data from NIST Open Machine Translation Evaluation tasks. The training data contains 353,796 sentence pairs, 8.7M Chinese words and 10.4M English words. The NIST 2003 and 2005 test data are respectively taken as the development and test set. We performed GIZA++ (Och and Ney, 2003) and the *grow-diag-final-and* symmetrizing strategy (Koehn et al., 2007) to obtain word alignments. The Berkeley Language Modeling Toolkit, berkeleylm-1.0b3[5] (Pauls and Klein, 2011), was employed to train (1) a five-gram LM on the Xinhua portion of LDC English Gigaword corpus v3 (LDC2007T07) and (2) a tri-gram dependency LM on the English dependency structures of the training data. We report the translation quality using the case-insensitive BLEU-4 metric (Papineni et al., 2002).

3.2 Statistics of dependencies

We compare the similarity of the dependencies with each other, as shown in Table 2. Basically, we investigate (1) if two dependency graphs of one sentence share the same root word and (2) if the head of one word in one sentence are identical in two dependency graphs. In terms of root word comparison, we observe that MST and CCG share 87.3% of identical root words, caused by borrowing roots from MST to CCG. Then, it is interesting that Berkeley and PAS+syn share 74.8% of identical root words. Note that the Berkeley parser is trained on the Penn treebank (Marcus et al., 1994) yet the HPSG parser is trained on the HPSG treebank (Miyao and Tsujii,

[5]http://code.google.com/p/berkeleylm/

Dependency	Precision	Recall	BLEU-Dev	BLEU-Test	# phrases	# hier rules	# illegal dep trees	# directed cycles
Moses-1	-	-	0.3349	0.3207	5.4M	-	-	-
Moses-2	-	-	0.3445	0.3262	0.7M	4.5M	-	-
MST	0.744	0.750	**0.3520**	0.3291	2.4M	2.1M	251	0
Malt	0.732	0.738	0.3423	0.3203	1.5M	1.3M	130,960	0
Berkeley	0.800	0.806	0.3475	0.3312	2.4M	2.2M	282	0
PAS+syn	**0.818**	**0.824**	0.3499	**0.3376**	2.2M	1.9M	10,411	5,853
PAS+sem	0.777	0.782	0.3484	0.3343	2.1M	1.6M	14,271	9,747
CCG	0.701	0.705	0.3442	0.3283	1.7M	1.3M	61,015	49,955

Table 3: Comparison of dependency and translation accuracies. Moses-1 = phrasal, Moses-2 = hierarchical.

	Malt	Berkeley	PAS +syn	PAS +sem	CCG
MST	70.5 (**77.3**)	62.5 (64.6)	69.2 (58.5)	53.3 (58.1)	**87.3** (61.7)
Malt		66.2 (63.2)	73.0 (57.7)	46.8 (56.6)	62.9 (58.1)
Berkeley			**74.8** (64.3)	44.2 (56.0)	56.5 (59.2)
PAS+ syn				59.3 (**79.1**)	62.9 (61.0)
PAS+ sem					60.0 (58.8)

Table 2: Comparison of the dependencies of the English sentences in the training data. Without () = % of similar root words; with () = % of similar head words.

2008). In terms of head word comparison, PAS+syn and PAS+sem share 79.1% of identical head words. This is basically due to that we used the similar PASs of the HPSG trees. Interestingly, there are only 59.3% identical root words shared by PAS+syn and PAS+sem. This reflects the significant difference between syntactic and semantic heads.

We also manually created the golden dependency trees for the first 200 English sentences in the training data. The precision/recall (P/R) are shown in Table 3. We observe that (1) the translation accuracies approximately follow the P/R scores yet are not that sensitive to their large variances, and (2) it is still tough for domain-adapting from the treebank-trained parsers to parse the real-world SMT data. PAS+syn performed the best by avoiding the errors of missing of arguments for a predicate, wrongly identified head words for a linguistic phrase, and inconsistency dependencies inside relatively long coordinate structures. These errors significantly influence the number of extractable translation rules and the final translation accuracies.

Note that, these P/R scores on the first 200 sentences (all from less than 20 newswire documents) shall only be taken as an approximation of the total training data and not necessarily exactly follow the tendency of the final BLEU scores. For example, CCG is worse than Malt in terms of P/R yet with a higher BLEU score. We argue this is mainly due to that the number of illegal dependency trees generated by Malt is the highest. Consequently, the number of flat/hierarchical rules generated by using Malt trees is the lowest. Also, PAS+sem has a lower P/R than Berkeley, yet their final BLEU scores are not statistically different.

3.3 Results

Table 3 also shows the BLEU scores, the number of flat phrases and hierarchical rules (both integrated with target dependency structures), and the number of illegal dependency trees generated by each parser. From the table, we have the following observations: (1) all the dependency structures (except Malt) achieved a significant better BLEU score than the phrasal Moses; (2) PAS+syn performed the best in the test set (0.3376), and it is significantly better than phrasal/hierarchical Moses ($p < 0.01$), MST ($p < 0.05$), Malt ($p < 0.01$), Berkeley ($p < 0.05$), and CCG ($p < 0.05$); and (3) CCG performed as well as MST and Berkeley. These results lead us to argue that the robustness of deep syntactic parsers can be advantageous in SMT compared with traditional dependency parsers.

4 Conclusion

We have constructed a string-to-dependency translation platform for comparing non-isomorphic target dependency structures. Specially, we proposed an algorithm for generating word-based dependency trees from PASs which are generated by a state-of-the-art HPSG parser. We found that dependency trees transformed from these HPSG PASs achieved the best dependency/translation accuracies.

Acknowledgments

We thank the anonymous reviewers for their constructive comments and suggestions.

References

Sabine Buchholz and Erwin Marsi. 2006. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City, June. Association for Computational Linguistics.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain, July. Association for Computational Linguistics.

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 NTCIR-9*, pages 559–578.

Richard Johansson and Pierre Nugues. 2007. Extended constituent-to-dependency conversion for english. In *In Proceedings of NODALIDA*, Tartu, Estonia, April.

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

Jeff Ma and Spyros Matsoukas. 2011. Bbn's systems for the chinese-english sub-task of the ntcir-9 patentmt evaluation. In *Proceedings of NTCIR-9*, pages 579–584.

David Magerman. 1995. Statistical decision-tree models for parsing. In *In Proceedings of of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 276–283.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Proceedings of the Workshop on HLT*, pages 114–119, Plainsboro.

Ryan McDonald and Joakim Nivre. 2011. Analyzing and integrating dependency parsers. *Computational Linguistics*, 37(1):197–230.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature forest models for probabilistic hpsg parsing. *Computational Lingustics*, 34(1):35–80.

Yusuke Miyao, Takashi Ninomiya, and Jun'ichi Tsujii. 2003. Probabilistic modeling of argument structures including non-local dependencies. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, pages 285–291, Borovets.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT*, pages 149–160.

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 *Proceedings of ACL*, pages 311–318.

Adam Pauls and Dan Klein. 2011. Faster and smaller n-gram language models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 258–267, Portland, Oregon, USA, June. Association for Computational Linguistics.

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, Rochester, New York, April. Association for Computational Linguistics.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

Ivan A. Sag, Thomas Wasow, and Emily M. Bender. 2003. *Syntactic Theory: A Formal Introduction*. Number 152 in CSLI Lecture Notes. CSLI Publications.

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.

Robust Conversion of CCG Derivations to Phrase Structure Trees

Jonathan K. Kummerfeld[†] **Dan Klein**[†] **James R. Curran**[‡]
[†]Computer Science Division
University of California, Berkeley
Berkeley, CA 94720, USA
{jkk,klein}@cs.berkeley.edu

[‡]ə-lab, School of IT
University of Sydney
Sydney, NSW 2006, Australia
james@it.usyd.edu.au

Abstract

We propose an improved, bottom-up method for converting CCG derivations into PTB-style phrase structure trees. In contrast with past work (Clark and Curran, 2009), which used simple transductions on category pairs, our approach uses richer transductions attached to single categories. Our conversion preserves more sentences under round-trip conversion (51.1% vs. 39.6%) and is more robust. In particular, unlike past methods, ours does not require ad-hoc rules over non-local features, and so can be easily integrated into a parser.

1 Introduction

Converting the Penn Treebank (PTB, Marcus et al., 1993) to other formalisms, such as HPSG (Miyao et al., 2004), LFG (Cahill et al., 2008), LTAG (Xia, 1999), and CCG (Hockenmaier, 2003), is a complex process that renders linguistic phenomena in formalism-specific ways. Tools for reversing these conversions are desirable for downstream parser use and parser comparison. However, reversing conversions is difficult, as corpus conversions may lose information or smooth over PTB inconsistencies.

Clark and Curran (2009) developed a CCG to PTB conversion that treats the CCG derivation as a phrase structure tree and applies hand-crafted rules to every pair of categories that combine in the derivation. Because their approach does not exploit the generalisations inherent in the CCG formalism, they must resort to ad-hoc rules over non-local features of the CCG constituents being combined (when a fixed pair of CCG categories correspond to multiple PTB structures). Even with such rules, they correctly convert only 39.6% of gold CCGbank derivations.

Our conversion assigns a set of bracket instructions to each word based on its CCG category, then follows the CCG derivation, applying and combining instructions at each combinatory step to build a phrase structure tree. This requires specific instructions for each category (not all pairs), and generic operations for each combinator. We cover all categories in the development set and correctly convert 51.1% of sentences. Unlike Clark and Curran's approach, we require no rules that consider non-local features of constituents, which enables the possibility of simple integration with a CKY-based parser.

The most common errors our approach makes involve nodes for clauses and rare spans such as QPs, NXs, and NACs. Many of these errors are inconsistencies in the original PTB annotations that are not recoverable. These issues make evaluating parser output difficult, but our method does enable an improved comparison of CCG and PTB parsers.

2 Background

There has been extensive work on converting parser output for evaluation, e.g. Lin (1998) and Briscoe et al. (2002) proposed using underlying dependencies for evaluation. There has also been work on conversion to phrase structure, from dependencies (Xia and Palmer, 2001; Xia et al., 2009) and from lexicalised formalisms, e.g. HPSG (Matsuzaki and Tsujii, 2008) and TAG (Chiang, 2000; Sarkar, 2001). Our focus is on CCG to PTB conversion (Clark and Curran, 2009).

2.1 Combinatory Categorial Grammar (CCG)

The lower half of Figure 1 shows a CCG derivation (Steedman, 2000) in which each word is assigned a *category*, and *combinatory rules* are applied to adjacent categories until only one remains. Categories

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 105–109,
Jeju, Republic of Korea, 8-14 July 2012.

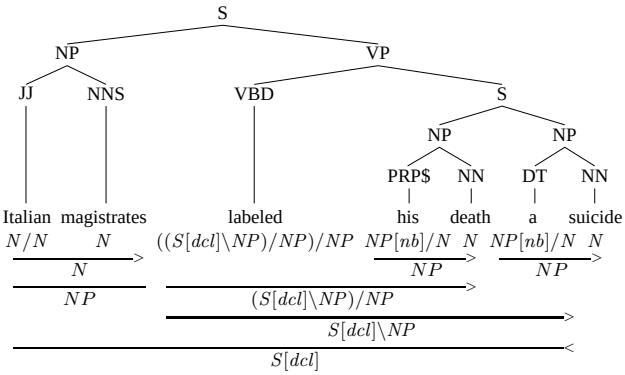


Figure 1: A crossing constituents example: *his . . . suicide* (PTB) crosses *labeled . . . death* (CCGbank).

Categories	Schema
N	create an NP
$((S[dcl]\backslash NP)/NP)/NP$	create a VP
N/N + N	place left under right
$NP[nb]/N$ + N	place left under right
$((S[dcl]\backslash NP)/NP)/NP$ + NP	place right under left
$(S[dcl]\backslash NP)/NP$ + NP	place right under left
NP + $S[dcl]\backslash NP$	place both under S

Table 1: Example C&C-CONV lexical and rule schemas.

can be atomic, e.g. the N assigned to *magistrates*, or complex functions of the form *result / arg*, where *result* and *arg* are categories and the slash indicates the argument's directionality. Combinators define how adjacent categories can combine. Figure 1 uses *function application*, where a complex category consumes an adjacent argument to form its result, e.g. $S[dcl]\backslash NP$ combines with the NP to its left to form an $S[dcl]$. More powerful combinators allow categories to combine with greater flexibility.

We cannot form a PTB tree by simply relabeling the categories in a CCG derivation because the mapping to phrase labels is many-to-many, CCG derivations contain extra brackets due to binarisation, and there are cases where the constituents in the PTB tree and the CCG derivation cross (e.g. in Figure 1).

2.2 Clark and Curran (2009)

Clark and Curran (2009), hereafter C&C-CONV, assign a *schema* to each leaf (lexical category) and rule (pair of combining categories) in the CCG derivation. The PTB tree is constructed from the CCG bottom-up, creating leaves with lexical schemas, then merging/adding sub-trees using rule schemas at each step.

The schemas for Figure 1 are shown in Table 1. These apply to create NPs over *magistrates*, *death*, and *suicide*, and a VP over *labeled*, and then combine the trees by placing one under the other at each step, and finally create an S node at the root.

C&C-CONV has sparsity problems, requiring schemas for all valid pairs of categories — at a minimum, the 2853 unique category combinations found in CCGbank. Clark and Curran (2009) create schemas for only 776 of these, handling the remainder with approximate catch-all rules.

C&C-CONV only specifies one simple schema for each rule (pair of categories). This appears reasonable at first, but frequently causes problems, e.g.:

$(N/N)/(N/N)$ + N/N
"more than" + "30" (1)
"relatively" + "small" (2)

Here either a QP bracket (1) or an ADJP bracket (2) should be created. Since both examples involve the same rule schema, C&C-CONV would incorrectly process them in the same way. To combat the most glaring errors, C&C-CONV manipulates the PTB tree with ad-hoc rules based on non-local features over the CCG nodes being combined — an approach that cannot be easily integrated into a parser.

These disadvantages are a consequence of failing to exploit the generalisations that CCG combinators define. We return to this example below to show how our approach handles both cases correctly.

3 Our Approach

Our conversion assigns a set of instructions to each lexical category and defines generic operations for each combinator that combine instructions. Figure 2 shows a typical instruction, which specifies the node to create and where to place the PTB trees associated with the two categories combining. More complex operations are shown in Table 2. Categories with multiple arguments are assigned one instruction per argument, e.g. *labeled* has three. These are applied one at a time, as each combinatory step occurs.

For the example from the previous section we begin by assigning the instructions shown in Table 3. Some of these can apply immediately as they do not involve an argument, e.g. *magistrates* has (NP f).

One of the more complex cases in the example is *Italian*, which is assigned (NP f {a}). This creates a new bracket, inserts the functor's tree, and flattens and inserts the argument's tree, producing:

(NP (JJ Italian) (NNS magistrates))

Figure 2: An example function application. Top row: CCG rule. Bottom row: applying instruction (VP f a).

Symbol	Meaning	Example
(X f a)	Add an X bracket around functor and argument	(VP f a)
{ }	Flatten enclosed node	(N f {a})
X*	Use same label as arg. or default to X	(S* f {a})
f_i	Place subtrees	(PP f_0 (S $f_{1..k}$ a))

Table 2: Types of operations in instructions.

For the complete example the final tree is almost correct but omits the S bracket around the final two NPs. To fix our example we could have modified our instructions to use the final symbol in Table 2. The subscripts indicate which subtrees to place where. However, for this particular construction the PTB annotations are inconsistent, and so we cannot recover without introducing more errors elsewhere.

For combinators other than function application, we combine the instructions in various ways. Additionally, we vary the instructions assigned based on the POS tag in 32 cases, and for the word *not*, to recover distinctions not captured by CCGbank categories alone. In 52 cases the later instructions depend on the structure of the argument being picked up. We have sixteen special cases for non-combinatory binary rules and twelve special cases for non-combinatory unary rules.

Our approach naturally handles our QP vs. ADJP example because the two cases have different lexical categories: $((N/N)/(N/N))\backslash(S[adj]\backslash NP)$ on *than* and $(N/N)/(N/N)$ on *relatively*. This lexical difference means we can assign different instructions to correctly recover the QP and ADJP nodes, whereas C&C-CONV applies the same schema in both cases as the categories combining are the same.

4 Evaluation

Using sections 00-21 of the treebanks, we handcrafted instructions for 527 lexical categories, a process that took under 100 hours, and includes all the categories used by the C&C parser. There are 647 further categories and 35 non-combinatory binary rules in sections 00-21 that we did not annotate. For unannotated categories, we use the instructions of the result category with an added instruction.

Category	Instruction set
N	(NP f)
N/N_1	(NP f {a})
$NP[nb]/N_1$	(NP f {a})
$((S[dcl]\backslash NP_3)/NP_2)/NP_1$	(VP f a)
	(VP {f} a)
	(S a f)

Table 3: Instruction sets for the categories in Figure 1.

System	Data	P	R	F	Sent.
C&C CONV	00 (all)	95.37	93.67	94.51	39.6
	00 (len ≤ 40)	95.85	94.39	95.12	42.1
	23 (all)	95.33	93.95	94.64	39.7
	23 (len ≤ 40)	95.44	94.04	94.73	41.9
This Work	00 (all)	96.69	96.58	96.63	51.1
	00 (len ≤ 40)	96.98	96.77	96.87	53.6
	23 (all)	96.49	96.11	**96.30**	**51.4**
	23 (len ≤ 40)	96.57	96.21	96.39	53.8

Table 4: PARSEVAL Precision, Recall, F-Score, and exact sentence match for converted gold CCG derivations.

Table 4 compares our approach with C&C-CONV on gold CCG derivations. The results shown are as reported by EVALB (Abney et al., 1991) using the Collins (1997) parameters. Our approach leads to increases on all metrics of at least 1.1%, and increases exact sentence match by over 11% (both absolute).

Many of the remaining errors relate to missing and extra clause nodes and a range of rare structures, such as QPs, NACs, and NXs. The only other prominent errors are single word spans, e.g. extra or missing ADVPs. Many of these errors are unrecoverable from CCGbank, either because inconsistencies in the PTB have been smoothed over or because they are genuine but rare constructions that were lost.

4.1 Parser Comparison

When we convert the output of a CCG parser, the PTB trees that are produced will contain errors created by our conversion as well as by the parser. In this section we are interested in comparing parsers, so we need to factor out errors created by our conversion.

One way to do this is to calculate a projected score (PROJ), as the parser result over the oracle result, but this is a very rough approximation. Another way is to evaluate only on the 51% of sentences for which our conversion from gold CCG derivations is perfect (CLEAN). However, even on this set our conversion

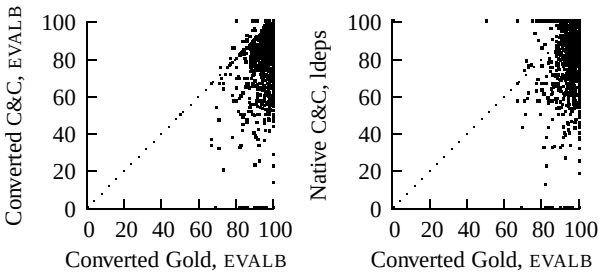


Figure 3: For each sentence in the treebank, we plot the converted parser output against gold conversion (left), and the original parser evaluation against gold conversion (right). Left: Most points lie below the diagonal, indicating that the quality of converted parser output (y) is upper bounded by the quality of conversion on gold parses (x). Right: No clear correlation is present, indicating that the set of sentences that are converted best (on the far right), are not necessarily easy to parse.

introduces errors, as the parser output may contain categories that are harder to convert.

Parser F-scores are generally higher on CLEAN, which could mean that this set is easier to parse, or it could mean that these sentences don't contain annotation inconsistencies, and so the parsers aren't incorrect for returning the true parse (as opposed to the one in the PTB). To test this distinction we look for correlation between conversion quality and parse difficulty on another metric. In particular, Figure 3 (right) shows CCG labeled dependency performance for the C&C parser vs. CCGbank conversion PARSEVAL scores. The lack of a strong correlation, and the spread on the line $x = 100$, supports the theory that these sentences are not necessarily easier to parse, but rather have fewer annotation inconsistencies.

In the left plot, the y-axis is PARSEVAL on converted C&C parser output. Conversion quality essentially bounds the performance of the parser. The few points above the diagonal are mostly short sentences on which the C&C parser uses categories that lead to one extra correct node. The main constructions on which parse errors occur, e.g. PP attachment, are rarely converted incorrectly, and so we expect the number of errors to be cumulative. Some sentences are higher in the right plot than the left because there are distinctions in CCG that are not always present in the PTB, e.g. the argument-adjunct distinction.

Table 5 presents F-scores for three PTB parsers and three CCG parsers (with their output converted by our method). One interesting comparison is between the PTB parser of Petrov and Klein (2007) and the CCG parser of Fowler and Penn (2010), which use the same underlying parser. The performance gap is partly due to structures in the PTB that are not recoverable from CCGbank, but probably also indicates that the split-merge model is less effective in CCG, which has far more symbols than the PTB.

Sentences	CLEAN	ALL	PROJ
Converted gold CCG			
CCGbank	100.0	96.3	–
Converted CCG			
Clark and Curran (2007)	90.9	85.5	88.8
Fowler and Penn (2010)	90.9	86.0	89.3
Auli and Lopez (2011)	91.7	86.2	89.5
Native PTB			
Klein and Manning (2003)	89.8	85.8	–
Petrov and Klein (2007)	93.6	90.1	–
Charniak and Johnson (2005)	94.8	91.5	–

Table 5: F-scores on section 23 for PTB parsers and CCG parsers with their output converted by our method. CLEAN is only on sentences that are converted perfectly from gold CCG (51%). ALL is over all sentences. PROJ is a projected F-score (ALL result / CCGbank ALL result).

It is difficult to make conclusive claims about the performance of the parsers. As shown earlier, CLEAN does not completely factor out the errors introduced by our conversion, as the parser output may be more difficult to convert, and the calculation of PROJ only roughly factors out the errors. However, the results do suggest that the performance of the CCG parsers is approaching that of the Petrov parser.

5 Conclusion

By exploiting the generalised combinators of the CCG formalism, we have developed a new method of converting CCG derivations into PTB-style trees. Our system, which is publicly available[1], is more effective than previous work, increasing exact sentence match by more than 11% (absolute), and can be directly integrated with a CCG parser.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful suggestions. This research was supported by a General Sir John Monash Fellowship, the Office of Naval Research under MURI Grant No. N000140911081, ARC Discovery grant DP1097291, and the Capital Markets CRC.

[1] http://code.google.com/p/berkeley-ccg2pst/

References

S. Abney, S. Flickenger, C. Gdaniec, C. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. Procedure for quantitatively comparing the syntactic coverage of english grammars. In *Proceedings of the workshop on Speech and Natural Language*, pages 306–311.

Michael Auli and Adam Lopez. 2011. A comparison of loopy belief propagation and dual decomposition for integrated ccg supertagging and parsing. In *Proceedings of ACL*, pages 470–480.

Ted Briscoe, John Carroll, Jonathan Graham, and Ann Copestake. 2002. Relational evaluation schemes. In *Proceedings of the Beyond PARSEVAL Workshop at LREC*, pages 4–8.

Aoife Cahill, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith, and Andy Way. 2008. Wide-coverage deep statistical parsing using automatic dependency structure annotation. *Computational Linguistics*, 34(1):81–124.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, pages 173–180.

David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of ACL*, pages 456–463.

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 and James R. Curran. 2009. Comparing the accuracy of CCG and penn treebank parsers. In *Proceedings of ACL*, pages 53–56.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL*, pages 16–23.

Timothy A. D. Fowler and Gerald Penn. 2010. Accurate context-free parsing with combinatory categorial grammar. In *Proceedings of ACL*, pages 335–344.

Julia Hockenmaier. 2003. *Data and models for statistical parsing with Combinatory Categorial Grammar*. Ph.D. thesis, School of Informatics, The University of Edinburgh.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of ACL*, pages 423–430.

Dekang Lin. 1998. A dependency-based method for evaluating broad-coverage parsers. *Natural Language Engineering*, 4(2):97–114.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330.

Takuya Matsuzaki and Jun'ichi Tsujii. 2008. Comparative parser performance analysis across grammar frameworks through automatic tree conversion using synchronous grammars. In *Proceedings of Coling*, pages 545–552.

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 *Proceedings of IJCNLP*, pages 684–693.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL*, pages 404–411.

Anoop Sarkar. 2001. Applying co-training methods to statistical parsing. In *Proceedings of NAACL*, pages 1–8.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press.

Fei Xia and Martha Palmer. 2001. Converting dependency structures to phrase structures. In *Proceedings of HLT*, pages 1–5.

Fei Xia, Owen Rambow, Rajesh Bhatt, Martha Palmer, and Dipti Misra Sharma. 2009. Towards a multi-representational treebank. In *Proceedings of the 7th International Workshop on Treebanks and Linguistic Theories*, pages 159–170.

Fei Xia. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 398–403.

Estimating Compact Yet Rich Tree Insertion Grammars

Elif Yamangil and **Stuart M. Shieber**
Harvard University
Cambridge, Massachusetts, USA
{elif, shieber}@seas.harvard.edu

Abstract

We present a Bayesian nonparametric model for estimating tree insertion grammars (TIG), building upon recent work in Bayesian inference of tree substitution grammars (TSG) via Dirichlet processes. Under our general variant of TIG, grammars are estimated via the Metropolis-Hastings algorithm that uses a context free grammar transformation as a proposal, which allows for cubic-time string parsing as well as tree-wide joint sampling of derivations in the spirit of Cohn and Blunsom (2010). We use the Penn treebank for our experiments and find that our proposal Bayesian TIG model not only has competitive parsing performance but also finds compact yet linguistically rich TIG representations of the data.

1 Introduction

There is a deep tension in statistical modeling of grammatical structure between providing good expressivity — to allow accurate modeling of the data with sparse grammars — and low complexity — making induction of the grammars and parsing of novel sentences computationally practical. Recent work that incorporated Dirichlet process (DP) nonparametric models into TSGs has provided an efficient solution to the problem of segmenting training data trees into elementary parse tree fragments to form the grammar (Cohn et al., 2009; Cohn and Blunsom, 2010; Post and Gildea, 2009). DP inference tackles this problem by exploring the space of all possible segmentations of the data, in search for fragments that are on the one hand large enough so that they incorporate the useful dependencies, and on the other small enough so that they recur and have a chance to be useful in analyzing unseen data.

The elementary trees combined in a TSG are, intuitively, primitives of the language, yet certain linguistic phenomena (notably various forms of modification) "split them up", preventing their reuse, leading to less sparse grammars than might be ideal. For instance, imagine modeling the following set of structures:

- [$_{NP}$ the [$_{NN}$ [$_{NN}$ [$_{NN}$ president] of the university] who resigned yesterday]]
- [$_{NP}$ the [$_{NN}$ former [$_{NN}$ [$_{NN}$ president] of the university]]]
- [$_{NP}$ the [$_{NN}$ [$_{NN}$ president] who resigned yesterday]]

A natural recurring structure here would be the structure "[$_{NP}$ the [$_{NN}$ president]]", yet it occurs not at all in the data.

TSGs are a special case of the more flexible grammar formalism of tree adjoining grammar (TAG) (Joshi et al., 1975). TAG augments TSG with an *adjunction operator* and a set of *auxiliary trees* in addition to the substitution operator and initial trees of TSG, allowing for "splicing in" of syntactic fragments within trees. In the example, by augmenting a TSG with an operation of adjunction, a grammar that hypothesizes auxiliary trees corresponding to adjoining "[$_{NN}$ former NN]", "[$_{NN}$ NN of the university]", and "[$_{NN}$ NN who resigned yesterday]" would be able to reuse the basic structure "[$_{NP}$ the [$_{NN}$ president]]".

Unfortunately, TAG's expressivity comes at the cost of greatly increased complexity. Parsing complexity for unconstrained TAG scales as $O(n^6)$, im-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 110–114,
Jeju, Republic of Korea, 8-14 July 2012.

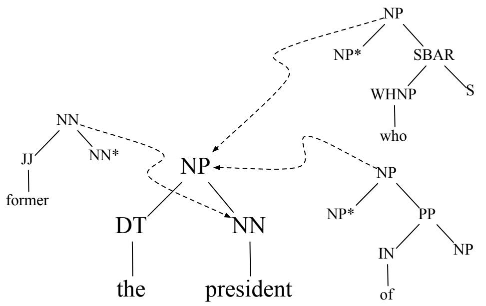


Figure 1: Example TIG derivation of an NP constituent: One left insertion (at NN) and two simultaneous right insertions (at NP).

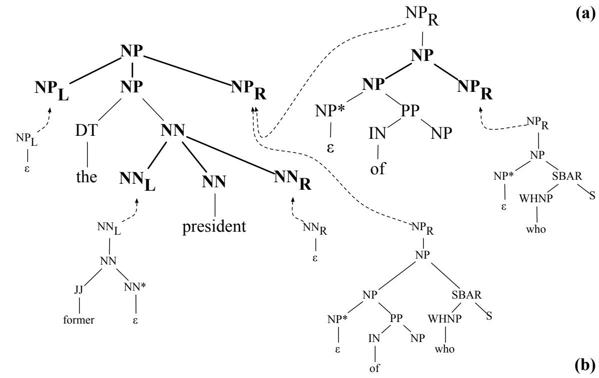


Figure 2: **TIG-to-TSG transform:** (a) and (b) illustrate transformed TSG derivations for two different TIG derivations of the same parse tree structure. The TIG nodes where we illustrate the transformation are in bold. (We suppress the rest of the transformational nodes.)

practical as compared to CFG and TSG's $O(n^3)$. In addition, the model selection problem for TAG is significantly more complicated than for TSG since one must reason about many more combinatorial options with two types of derivation operators.[1] This has led researchers to resort to heuristic grammar extraction techniques (Chiang, 2000; Carreras et al., 2008) or using a very small number of grammar categories (Hwa, 1998).

Hwa (1998) first proposed to use tree-insertion grammars (TIG), a kind of expressive compromise between TSG and TAG, as a substrate on which to build grammatical inference. TIG constrains the adjunction operation so that spliced-in material falls completely to the left or completely to the right of the splice point. By restricting the form of possible auxiliary trees to only *left* or *right* auxiliary trees in this way, TIG remains within the realm of context-free formalisms (with cubic complexity) while still modeling rich linguistic phenomena (Schabes and Waters, 1995). Figure 1 depicts some examples of TIG derivations.

Sharing the same intuitions, Shindo et al. (2011) have provided a previous attempt at combining TIG and Bayesian nonparametric principles, albeit with severe limitations. Their TIG variant (which we will refer to as TIG_0) is highly constrained in the following ways.

1. The foot node in an auxiliary tree must be the immediate child of the root node.
2. Only one adjunction can occur at a given node.
3. Even modeling multiple adjunction with root adjunction is disallowed. There is thus no recursion possibility with adjunction, no stacking of auxiliary trees.
4. As a consequence of the prior two constraints, no adjunction along the spines of auxiliary trees is allowed.
5. As a consequence of the first constraint, all nonterminals along the spine of an auxiliary tree are identical.

In this paper we explore a Bayesian nonparametric model for estimating a far more expressive version of TIG, and compare its performance against TSG and the restricted TIG_0 variant. Our more general formulation avoids these limitations by supporting the following features and thus relaxing four of the five restrictions of TIG_0.

1. Auxiliary trees may have the foot node at depth greater than one.[2]
2. Both left and right adjunctions may occur at the same node.
3. Simultanous adjunction (that is, more than one left or right adjunction per node) is allowed via root adjunction.
4. Adjunctions may occur along the spines of auxiliary trees.

The increased expressivity of our TIG variant is motivated both linguistically and practically. From a linguistic point of view: Deeper auxiliary trees can help model large patterns of insertion and potential correlations between lexical items that extend over multiple levels of tree. Combining left and right auxiliary trees can help model modifiers of the same node from left and right (combination of adjectives

[1] This can be seen by the fact that tree-path languages under TAG are context free, whereas they are regular for TSG. (Schabes and Waters, 1995)

[2] Throughout the paper, we will refer to the depth of an auxiliary tree to indicate the length of its spine.

and relative clauses for instance). Simultaneous insertion allows us to deal with multiple independent modifiers for the same constituent (for example, a series of adjectives). From a practical point of view, we show that an induced TIG provides modeling performance superior to TSG and comparable with TIG_0. However we show that the grammars we induce are *compact yet rich*, in that they succinctly represent complex linguistic structures.

2 Probabilistic Model

In the basic nonparametric TSG model, there is an independent DP for every grammar category (such as $c = NP$), each of which uses a base distribution P_0 that generates an initial tree by making stepwise decisions.

$$G_c^{\text{init}} \sim \text{DP}(\alpha_c^{\text{init}}, P_0^{\text{init}}(\cdot \mid c))$$

The canonical P_0 uses a probabilistic CFG $\tilde{P}$ that is fixed a priori to sample CFG rules top-down and Bernoulli variables for determining where substitutions should occur (Cohn et al., 2009; Cohn and Blunsom, 2010).

We extend this model by adding specialized DPs for left and right auxiliary trees.[3]

$$G_c^{\text{right}} \sim \text{DP}(\alpha_c^{\text{right}}, P_0^{\text{right}}(\cdot \mid c))$$

Therefore, we have an exchangeable process for generating right auxiliary trees

$$p(a_j \mid \mathbf{a}_{<j}) = \frac{n_{a_j} + \alpha_c^{\text{right}} P_0^{\text{right}}(a_j \mid c)}{j - 1 + \alpha_c^{\text{right}}} \quad (1)$$

as for initial trees in TSG.

We must define three distinct base distributions for initial trees, left auxiliary trees, and right auxiliary trees. P_0^{init} generates an initial tree with root label c by sampling CFG rules from $\tilde{P}$ and making a binary decision at every node generated whether to leave it as a frontier node or further expand (with probability β_c) (Cohn et al., 2009). Similarly, our P_0^{right} generates a right auxiliary tree with root label c by first making a binary decision whether to generate an immediate foot or not (with probability γ_c^{right}), and then sampling an appropriate CFG rule from $\tilde{P}$. For the right child, we sample an initial tree from P_0^{init}. For the left child, if decision to generate an immediate foot was made, we generate a foot node, and stop. Otherwise we recur into P_0^{right} which generates a right auxiliary tree that becomes the left child.

[3] We use right insertions for illustration; the symmetric analog applies to left insertions.

```
(VP (, ,) (VP PP (VP (, ,) VP*)))
(VP (SBAR (WHADVP (WRB (WRB When) ) ) S) (VP (, ,) VP*))
(VP (PP (IN For) (NP NN )) (VP (, ,) VP*))
(VP (CC But) (VP PP (VP (, ,) VP*)))
(VP ADVP (VP (, ,) VP*))
(IN (ADVP (RB (RB particularly) ) ) IN*)
(NP PP (NP (CC and) (NP PP NP*)))
```

Figure 3: Example left auxiliary trees that occur in the top derivations for Section 23. Simultaneous insertions occur most frequently for the labels VP (85 times), NNS (21 times), NNP (14 times).

We bring together these three sets of processes via a set of insertion parameters $\mu_c^{\text{left}}, \mu_c^{\text{right}}$. In any derivation, for every initial tree node labelled c (except for frontier nodes) we determine whether or not there are insertions at this node by sampling a Bernoulli(μ_c^{left}) distributed left insertion variable and a Bernoulli(μ_c^{right}) distributed right insertion variable. For left auxiliary trees, we treat the nodes that are *not* along the spine of the auxiliary tree the same way we treat initial tree nodes, however for nodes that are along the spine (including root nodes, excluding foot nodes) we consider only left insertions by sampling the left insertion variable (symmetrically for right insertions).

3 Inference

Given this model, our inference task is to explore optimal derivations underlying the data. Since TIG derivations are highly structured objects, a basic sampling strategy based on local node-level moves such as Gibbs sampling (Geman and Geman, 1984) would not hold much promise. Following previous work, we design a blocked Metropolis-Hastings sampler that samples derivations per entire parse trees all at once in a joint fashion (Cohn and Blunsom, 2010; Shindo et al., 2011). This is achieved by proposing derivations from an approximating distribution and stochastically correcting via accept/reject to achieve convergence into the correct posterior (Johnson et al., 2007).

Since our base distributions factorize over levels of tree, CFG is the most convenient choice for a

CFG rule	CFG probability
	Base distribution: P_0^{init}
$\text{NP} \rightarrow \text{NP}^{\text{init}}$	$\alpha_c^{\text{init}} / (n_{\text{NP}}^{\text{init}} + \alpha_c^{\text{init}})$
$\text{NP}^{\text{init}} \rightarrow \text{NP}_{\text{L}}$ _NP^{init} NP_{R}	1.0
_$\text{NP}^{\text{init}} \rightarrow$ DT NN	$\tilde{P}(\text{NP} \rightarrow \text{DT NN}) \times (1 - \beta_{\text{DT}}) \times (1 - \beta_{\text{NN}})$
_$\text{NP}^{\text{init}} \rightarrow$ DT NN^{init}	$\tilde{P}(\text{NP} \rightarrow \text{DT NN}) \times (1 - \beta_{\text{DT}}) \times \beta_{\text{NN}}$
_$\text{NP}^{\text{init}} \rightarrow \text{DT}^{\text{init}}$ NN	$\tilde{P}(\text{NP} \rightarrow \text{DT NN}) \times \beta_{\text{DT}} \times (1 - \beta_{\text{NN}})$
_$\text{NP}^{\text{init}} \rightarrow \text{DT}^{\text{init}}$ NN^{init}	$\tilde{P}(\text{NP} \rightarrow \text{DT NN}) \times \beta_{\text{DT}} \times \beta_{\text{NN}}$
	Base distribution: P_0^{right}
$\text{NP}_{\text{R}} \rightarrow \text{NP}^{\text{right}}$	$\mu_{\text{NP}}^{\text{right}} \times \left(\alpha_c^{\text{right}} / (n_{\text{NP}}^{\text{right}} + \alpha_c^{\text{right}})\right)$
$\text{NP}_{\text{R}} \rightarrow \epsilon$	$1 - \mu_{\text{NP}}^{\text{right}}$
$\text{NP}^{\text{right}} \rightarrow$ _NP^{right} NP_{R}	1.0
_$\text{NP}^{\text{right}} \rightarrow \text{NP}*$ $\text{SBAR}^{\text{init}}$	$\tilde{P}(\text{NP} \rightarrow \text{NP SBAR} \mid \text{NP} \rightarrow \text{NP }\_) \times (1 - \gamma_{\text{NP}}^{\text{right}}) \times (1 - \beta_{\text{SBAR}})$
_$\text{NP}^{\text{right}} \rightarrow \text{NP}*$ SBAR	$\tilde{P}(\text{NP} \rightarrow \text{NP SBAR} \mid \text{NP} \rightarrow \text{NP }\_) \times (1 - \gamma_{\text{NP}}^{\text{right}}) \times \beta_{\text{SBAR}}$
_$\text{NP}^{\text{right}} \rightarrow \text{NP}^{\text{right}}$ $\text{SBAR}^{\text{init}}$	$\tilde{P}(\text{NP} \rightarrow \text{NP SBAR} \mid \text{NP} \rightarrow \text{NP }\_) \times \gamma_{\text{NP}}^{\text{right}} \times (1 - \beta_{\text{SBAR}})$
_$\text{NP}^{\text{right}} \rightarrow \text{NP}^{\text{right}}$ SBAR	$\tilde{P}(\text{NP} \rightarrow \text{NP SBAR} \mid \text{NP} \rightarrow \text{NP }\_) \times \gamma_{\text{NP}}^{\text{right}} \times \beta_{\text{SBAR}}$

Figure 4: Transformation CFG rules that represent infinite base distributions. P_0^{init} is taken from Cohn and Blunsom (2010). Underscored labels (such as _NP^{right} as opposed to NP^{right}) are used to differentiate the pre-insertion nodes in Figure 2 from the post-insertion ones. P_0^{left} rules are omitted for brevity and mirror the P_0^{right} rules above.

Model	FMeasure	# Initial Trees	# Auxiliary Trees (# Left)
TSG	77.51	6.2K	-
TIG_0	78.46	6.0K	251 (137)
TIG	78.62	5.6K	604 (334)

Figure 5: EVALB results after training on Section 2 and testing on Section 23. Note that TIG finds a compact yet rich representation. Elementary tree counts are based on ones with count > 1.

proposal distribution. Fortunately, Schabes and Waters (1995) provide an (exact) transformation from a fully general TIG into a TSG that generates the same string languages. It is then straightforward to represent this TSG as a CFG using the Goodman transform (Goodman, 2002; Cohn and Blunsom, 2010). Figure 4 lists the additional CFG productions we have designed, as well as the rules used that trigger them.

4 Evaluation Results

We use the standard Penn treebank methodology of training on sections 2–21 and testing on section 23. All our data is head-binarized and words occurring only once are mapped into unknown categories of the Berkeley parser. As has become standard, we carried out a small treebank experiment where we train on Section 2, and a large one where we train on the full training set. All hyperparameters are resampled under appropriate vague gamma and beta priors. All reported numbers are averages over three runs. Parsing results are based on the maximum probability parse which was obtained by sampling derivations under the transform CFG.

We compare our system (referred to as TIG) to our implementation of the TSG system of (Cohn and Blunsom, 2010) (referred to as TSG) and the constrained TIG variant of (Shindo et al., 2011) (referred to as TIG_0). The upshot of our experiments is that, while on the large training set all models have similar performance ($85.6, 85.3, 85.4$ for TSG, TIG_0 and TIG respectively), on the small dataset insertion helps nonparametric model to find more compact and generalizable representations for the data, which affects parsing performance (Figure 4). Although TIG_0 has performance close to TIG, note that TIG achieves this performance using a more succinct representation and extracting a rich set of auxiliary trees. As a result, TIG finds many chances to apply insertions to test sentences, whereas TIG_0 depends mostly on TSG rules. If we look at the most likely derivations for the test data, TIG_0 assigns 663 insertions (351 left insertions) in the parsing of entire Section 23, meanwhile TIG assigns 3924 (2100 left insertions). Some of these linguistically sophisticated auxiliary trees that apply to test data are listed in Figure 3.

5 Conclusion

We described a nonparametric Bayesian inference scheme for estimating TIG grammars and showed the power of TIG formalism over TSG for returning rich, generalizable, yet compact representations of data. The nonparametric inference scheme presents a principled way of addressing the difficult model selection problem with TIG which has been prohibitive in this area of research. TIG still remains within context free and both our sampling and parsing techniques are highly scalable.

Acknowledgements

The first author was supported in part by a Google PhD Fellowship in Natural Language Processing.

References

Xavier Carreras, Michael Collins, and Terry Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 9–16, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Chiang. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 456–463, Morristown, NJ, USA. Association for Computational Linguistics.

Trevor Cohn and Phil Blunsom. 2010. Blocked inference in Bayesian tree substitution grammars. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 225–230, Stroudsburg, PA, USA. Association for Computational Linguistics.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing compact but accurate tree-substitution grammars. In *NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–556, Morristown, NJ, USA. Association for Computational Linguistics.

S. Geman and D. Geman. 1984. Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images. pages 6:721–741.

J. Goodman. 2002. Efficient parsing of DOP with PCFG-reductions. *Bod et al. 2003*.

Rebecca Hwa. 1998. An empirical evaluation of probabilistic lexicalized tree insertion grammars. In *Proceedings of the 17th international conference on Computational linguistics - Volume 1*, pages 557–563, Morristown, NJ, USA. Association for Computational Linguistics.

Mark Johnson, Thomas Griffiths, and Sharon Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 139–146, Rochester, New York, April. Association for Computational Linguistics.

Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136–163.

Matt Post and Daniel Gildea. 2009. Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48, Suntec, Singapore, August. Association for Computational Linguistics.

Remko Scha and Rens Bod. 2003. Efficient parsing of DOP with PCFG-reductions, October.

Yves Schabes and Richard C. Waters. 1995. Tree insertion grammar: a cubic-time parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Comput. Linguist.*, 21:479–513, December.

Hiroyuki Shindo, Akinori Fujino, and Masaaki Nagata. 2011. Insertion operator for Bayesian tree substitution grammars. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 206–211, Stroudsburg, PA, USA. Association for Computational Linguistics.

Topic Models for Dynamic Translation Model Adaptation

Vladimir Eidelman
Computer Science
and UMIACS
University of Maryland
College Park, MD
vlad@umiacs.umd.edu

Jordan Boyd-Graber
iSchool
and UMIACS
University of Maryland
College Park, MD
jbg@umiacs.umd.edu

Philip Resnik
Linguistics
and UMIACS
University of Maryland
College Park, MD
resnik@umd.edu

Abstract

We propose an approach that biases machine translation systems toward relevant translations based on topic-specific contexts, where topics are induced in an unsupervised way using topic models; this can be thought of as inducing subcorpora for adaptation without any human annotation. We use these topic distributions to compute topic-dependent lexical weighting probabilities and directly incorporate them into our translation model as features. Conditioning lexical probabilities on the topic biases translations toward topic-relevant output, resulting in significant improvements of up to 1 BLEU and 3 TER on Chinese to English translation over a strong baseline.

1 Introduction

The performance of a statistical machine translation (SMT) system on a translation task depends largely on the suitability of the available parallel training data. Domains (e.g., newswire vs. blogs) may vary widely in their lexical choices and stylistic preferences, and what may be preferable in a general setting, or in one domain, is not necessarily preferable in another domain. Indeed, sometimes the domain can change the meaning of a phrase entirely.

In a food related context, the Chinese sentence "粉丝很多 " ("fěnsī hěnduō") would mean "They have a lot of vermicelli"; however, in an informal Internet conversation, this sentence would mean "They have a lot of fans". Without the broader context, it is impossible to determine the correct translation in otherwise identical sentences.

This problem has led to a substantial amount of recent work in trying to bias, or adapt, the translation model (TM) toward particular domains of interest (Axelrod et al., 2011; Foster et al., 2010; Snover et al., 2008).[1] The intuition behind TM adaptation is to increase the likelihood of selecting relevant phrases for translation. Matsoukas et al. (2009) introduced assigning a pair of binary features to each training sentence, indicating sentences' *genre and collection* as a way to capture domains. They then learn a mapping from these features to sentence weights, use the sentence weights to bias the model probability estimates and subsequently learn the model weights. As sentence weights were found to be most beneficial for lexical weighting, Chiang et al. (2011) extends the same notion of conditioning on provenance (i.e., the origin of the text) by removing the separate mapping step, directly optimizing the weight of the genre and collection features by computing a separate word translation table for each feature, estimated from only those sentences that comprise that genre or collection.

The common thread throughout prior work is the concept of a *domain*. A domain is typically a hard constraint that is externally imposed and hand labeled, such as genre or corpus collection. For example, a sentence either comes from newswire, or weblog, but not both. However, this poses several problems. First, since a sentence contributes its counts only to the translation table for the source it came from, many word pairs will be unobserved for a given table. This sparsity requires smoothing. Second, we may not know the (sub)corpora our training

[1] Language model adaptation is also prevalent but is not the focus of this work.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 115–119,
Jeju, Republic of Korea, 8-14 July 2012.

data come from; and even if we do, "subcorpus" may not be the most useful notion of domain for better translations.

We take a finer-grained, flexible, unsupervised approach for lexical weighting by domain. We induce unsupervised domains from large corpora, and we incorporate soft, probabilistic domain membership into a translation model. Unsupervised modeling of the training data produces naturally occurring subcorpora, generalizing beyond corpus and genre. Depending on the model used to select subcorpora, we can bias our translation toward any arbitrary distinction. This reduces the problem to identifying what automatically defined subsets of the training corpus may be beneficial for translation.

In this work, we consider the underlying *latent topics* of the documents (Blei et al., 2003). Topic modeling has received some use in SMT, for instance Bilingual LSA adaptation (Tam et al., 2007), and the BiTAM model (Zhao and Xing, 2006), which uses a bilingual topic model for learning alignment. In our case, by building a topic distribution for the source side of the training data, we abstract the notion of domain to include automatically derived subcorpora with probabilistic membership. This topic model infers the topic distribution of a test set and biases sentence translations to appropriate topics. We accomplish this by introducing topic dependent lexical probabilities directly as features in the translation model, and interpolating them log-linearly with our other features, thus allowing us to discriminatively optimize their weights on an arbitrary objective function. Incorporating these features into our hierarchical phrase-based translation system significantly improved translation performance, by up to 1 BLEU and 3 TER over a strong Chinese to English baseline.

2 Model Description

Lexical Weighting Lexical weighting features estimate the quality of a phrase pair by combining the lexical translation probabilities of the words in the phrase[2] (Koehn et al., 2003). Lexical conditional probabilities $p(e|f)$ are obtained with maximum likelihood estimates from relative frequencies $c(f,e)/\sum_e c(f,e)$. Phrase pair probabilities $p(\overline{e}|\overline{f})$ are computed from these as described in Koehn et al. (2003).

Chiang et al. (2011) showed that is it beneficial to condition the lexical weighting features on provenance by assigning each sentence pair a set of features, $f_s(\overline{e}|\overline{f})$, one for each domain s, which compute a new word translation table $p_s(e|f)$ estimated from only those sentences which belong to s: $c_s(f,e)/\sum_e c_s(f,e)$, where $c_s(\cdot)$ is the number of occurrences of the word pair in s.

Topic Modeling for MT We extend provenance to cover a set of automatically generated topics z_n. Given a parallel training corpus T composed of documents d_i, we build a source side topic model over T, which provides a topic distribution $p(z_n|d_i)$ for $z_n = \{1, \ldots, K\}$ over each document, using Latent Dirichlet Allocation (LDA) (Blei et al., 2003). Then, we assign $p(z_n|d_i)$ to be the topic distribution for every sentence $x_j \in d_i$, thus enforcing topic sharing across sentence pairs in the same document instead of treating them as unrelated. Computing the topic distribution over a document and assigning it to the sentences serves to tie the sentences together in the document context.

To obtain the lexical probability conditioned on topic distribution, we first compute the expected count $e_{z_n}(e,f)$ of a word pair under topic z_n:

$$e_{z_n}(e,f) = \sum_{d_i \in T} p(z_n|d_i) \sum_{x_j \in d_i} c_j(e,f) \quad (1)$$

where $c_j(\cdot)$ denotes the number of occurrences of the word pair in sentence x_j, and then compute:

$$p_{z_n}(e|f) = \frac{e_{z_n}(e,f)}{\sum_e e_{z_n}(e,f)} \quad (2)$$

Thus, we will introduce $2{\cdot}K$ new word translation tables, one for each $p_{z_n}(e|f)$ and $p_{z_n}(f|e)$, and as many new corresponding features $f_{z_n}(\overline{e}|\overline{f})$, $f_{z_n}(\overline{f}|\overline{e})$. The actual feature values we compute will depend on the topic distribution of the document we are translating. For a test document V, we infer topic assignments on V, $p(z_n|V)$, keeping the topics found from T fixed. The feature value then becomes $f_{z_n}(\overline{e}|\overline{f}) = -\log\left\{p_{z_n}(\overline{e}|\overline{f}) \cdot p(z_n|V)\right\}$, a combination of the topic dependent lexical weight and the

[2]For hierarchical systems, these correspond to translation rules.

topic distribution of the sentence from which we are extracting the phrase. To optimize the weights of these features we combine them in our linear model with the other features when computing the model score for each phrase pair[3]:

$$\underbrace{\sum_{p} \lambda_p h_p(e,f)}_{\text{unadapted features}} + \underbrace{\sum_{z_n} \lambda_{z_n} f_{z_n}(\overline{e}|\overline{f})}_{\text{adapted features}} \quad (3)$$

Combining the topic conditioned word translation table $p_{z_n}(e|f)$ computed from the training corpus with the topic distribution $p(z_n|V)$ of the test sentence being translated provides a probability on how relevant that translation table is to the sentence. This allows us to bias the translation toward the topic of the sentence. For example, if topic k is dominant in T, $p_k(\overline{e}|\overline{f})$ may be quite large, but if $p(k|V)$ is very small, then we should steer away from this phrase pair and select a competing phrase pair which may have a lower probability in T, but which is more relevant to the test sentence at hand.

In many cases, document delineations may not be readily available for the training corpus. Furthermore, a document may be too broad, covering too many disparate topics, to effectively bias the weights on a phrase level. For this case, we also propose a local LDA model (LTM), which treats each sentence as a separate document.

While Chiang et al. (2011) has to *explicitly* smooth the resulting $p_s(e|f)$, since many word pairs will be unseen for a given domain s, we are already performing an *implicit* form of smoothing (when computing the expected counts), since each document has a distribution over all topics, and therefore we have some probability of observing each word pair in every topic.

Feature Representation After obtaining the topic conditional features, there are two ways to present them to the model. They could answer the question F_1: What is the probability under topic 1, topic 2, etc., or F_2: What is the probability under the most probable topic, second most, etc.

A model using F_1 learns whether a *specific* topic is useful for translation, i.e., feature f_1 would be $f_1 := p_{z=1}(\overline{e}|\overline{f}) \cdot p(z = 1|V)$. With F_2, we are learning how useful knowledge of the topic distribution is, i.e., $f_1 := p_{(\arg\max_{z_n}(p(z_n|V)))}(\overline{e}|\overline{f}) \cdot p(\arg\max_{z_n}(p(z_n|V))|V)$.

Using F_1, if we restrict our topics to have a one-to-one mapping with genre/collection[4] we see that our method fully recovers Chiang (2011).

F_1 is appropriate for *cross-domain* adaptation when we have advance knowledge that the distribution of the tuning data will match the test data, as in Chiang (2011), where they tune and test on web. In general, we may not know what our data will be, so this will overfit the tuning set.

F_2, however, is intuitively what we want, since we do not want to bias our system toward a specific distribution, but rather learn to utilize information from *any* topic distribution if it helps us create topic relevant translations. F_2 is useful for *dynamic* adaptation, where the adapted feature weight changes based on the source sentence.

Thus, F_2 is the approach we use in our work, which allows us to tune our system weights toward having topic information be useful, not toward a specific distribution.

3 Experiments

Setup To evaluate our approach, we performed experiments on Chinese to English MT in two settings. First, we use the FBIS corpus as our training bitext. Since FBIS has document delineations, we compare local topic modeling (LTM) with modeling at the document level (GTM). The second setting uses the non-UN and non-HK Hansards portions of the NIST training corpora with LTM only. Table 1 summarizes the data statistics. For both settings, the data were lowercased, tokenized and aligned using GIZA++ (Och and Ney, 2003) to obtain bidirectional alignments, which were symmetrized using the `grow-diag-final-and` method (Koehn et al., 2003). The Chinese data were segmented using the Stanford segmenter. We trained a trigram LM on the English side of the corpus with an additional 150M words randomly selected from the non-NYT and non-LAT portions of the Gigaword v4 corpus using modified Kneser-Ney smoothing (Chen and Goodman, 1996). We used `cdec` (Dyer et al.,

[3]The unadapted lexical weight $p(\overline{e}|\overline{f})$ is included in the model features.

[4]By having as many topics as genres/collections and setting $p(z_n|d_i)$ to 1 for every sentence in the collection and 0 to everything else.

Corpus	Sentences	Tokens	
		En	Zh
FBIS	269K	10.3M	7.9M
NIST	1.6M	44.4M	40.4M

Table 1: Corpus statistics

2010) as our decoder, and tuned the parameters of the system to optimize BLEU (Papineni et al., 2002) on the NIST MT06 tuning corpus using the Margin Infused Relaxed Algorithm (MIRA) (Crammer et al., 2006; Eidelman, 2012). Topic modeling was performed with Mallet (Mccallum, 2002), a standard implementation of LDA, using a Chinese stoplist and setting the per-document Dirichlet parameter $\alpha = 0.01$. This setting of was chosen to encourage sparse topic assignments, which make induced subdomains consistent within a document.

Results Results for both settings are shown in Table 2. GTM models the latent topics at the document level, while LTM models each sentence as a separate document. To evaluate the effect topic granularity would have on translation, we varied the number of latent topics in each model to be 5, 10, and 20. On FBIS, we can see that both models achieve moderate but consistent gains over the baseline on both BLEU and TER. The best model, LTM-10, achieves a gain of about 0.5 and 0.6 BLEU and 2 TER. Although the performance on BLEU for both the 20 topic models LTM-20 and GTM-20 is suboptimal, the TER improvement is better. Interestingly, the difference in translation quality between capturing document coherence in GTM and modeling purely on the sentence level is not substantial.[5] In fact, the opposite is true, with the LTM models achieving better performance.[6]

On the NIST corpus, LTM-10 again achieves the best gain of approximately 1 BLEU and up to 3 TER. LTM performs on par with or better than GTM, and provides significant gains even in the NIST data setting, showing that this method can be effectively applied directly on the sentence level to large training corpora which have no document markings. Depending on the diversity of training corpus, a varying number of underlying topics may be appropriate. However, in both settings, 10 topics performed best.

Model	MT03		MT05	
	↑BLEU	↓TER	↑BLEU	↓TER
BL	28.72	65.96	27.71	67.58
GTM-5	28.95^{ns}	65.45	27.98^{ns}	67.38^{ns}
GTM-10	29.22	64.47	**28.19**	66.15
GTM-20	29.19	**63.41**	28.00^{ns}	**64.89**
LTM-5	29.23	64.57	**28.19**	66.30
LTM-10	**29.29**	63.98	**28.18**	65.56
LTM-20	29.09^{ns}	63.57	27.90^{ns}	65.17
Model	MT03		MT05	
	↑BLEU	↓TER	↑BLEU	↓TER
BL	34.31	61.14	30.63	65.10
MERT	34.60	60.66	30.53	64.56
LTM-5	35.21	59.48	31.47	62.34
LTM-10	**35.32**	**59.16**	**31.56**	**62.01**
LTM-20	33.90^{ns}	60.89^{ns}	30.12^{ns}	63.87

Table 2: Performance using FBIS training corpus (top) and NIST corpus (bottom). Improvements are significant at the p <0.05 level, except where indicated (^{ns}).

4 Discussion and Conclusion

Applying SMT to new domains requires techniques to inform our algorithms how best to adapt. This paper extended the usual notion of domains to finer-grained topic distributions induced in an unsupervised fashion. We show that incorporating lexical weighting features conditioned on soft domain membership directly into our model is an effective strategy for dynamically biasing SMT towards relevant translations, as evidenced by significant performance gains. This method presents several advantages over existing approaches. We can construct a topic model once on the training data, and use it infer topics on any test set to adapt the translation model. We can also incorporate large quantities of additional data (whether parallel or not) in the source language to infer better topics without relying on collection or genre annotations. Multilingual topic models (Boyd-Graber and Resnik, 2010) would provide a technique to use data from multiple languages to ensure consistent topics.

[5] An avenue of future work would condition the sentence topic distribution on a document distribution over topics (Teh et al., 2006).

[6] As an empirical validation of our earlier intuition regarding feature representation, presenting the features in the form of F_1 caused the performance to remain virtually unchanged from the baseline model.

Acknowledgments

Vladimir Eidelman is supported by a National Defense Science and Engineering Graduate Fellowship. This work was also supported in part by NSF grant #1018625, ARL Cooperative Agreement W911NF-09-2-0072, and by the BOLT and GALE programs of the Defense Advanced Research Projects Agency, Contracts HR0011-12-C-0015 and HR0011-06-2-001, respectively. Any opinions, findings, conclusions, or recommendations expressed are the authors' and do not necessarily reflect those of the sponsors.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of Emperical Methods in Natural Language Processing*.

David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:2003.

Jordan Boyd-Graber and Philip Resnik. 2010. Holistic sentiment analysis across languages: Multilingual supervised latent Dirichlet allocation. In *Proceedings of Emperical Methods in Natural Language Processing*.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 310–318.

David Chiang, Steve DeNeefe, and Michael Pust. 2011. Two easy improvements to lexical weighting. In *Proceedings of the Human Language Technology Conference*.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

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 ACL System Demonstrations*.

Vladimir Eidelman. 2012. Optimization strategies for online large-margin learning in machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of Emperical Methods in Natural Language Processing*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. 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, Stroudsburg, PA, USA.

Spyros Matsoukas, Antti-Veikko I. Rosti, and Bing Zhang. 2009. Discriminative corpus weight estimation for machine translation. In *Proceedings of Emperical Methods in Natural Language Processing*.

A. K. Mccallum. 2002. MALLET: A Machine Learning for Language Toolkit.

Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. In *Computational Linguistics*, volume 29(21), pages 19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 311–318.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2008. Language and translation model adaptation using comparable corpora. In *Proceedings of Emperical Methods in Natural Language Processing*.

Yik-Cheung Tam, Ian Lane, and Tanja Schultz. 2007. Bilingual LSA-based adaptation for statistical machine translation. *Machine Translation*, 21(4):187–207.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Bing Zhao and Eric P. Xing. 2006. BiTAM: Bilingual topic admixture models for word alignment. In *Proceedings of the Association for Computational Linguistics*.

Detecting Semantic Equivalence and Information Disparity in Cross-lingual Documents

Yashar Mehdad **Matteo Negri** **Marcello Federico**
Fondazione Bruno Kessler, FBK-irst
Trento , Italy
{mehdad|negri|federico}@fbk.eu

Abstract

We address a core aspect of the multilingual content synchronization task: the identification of novel, more informative or semantically equivalent pieces of information in two documents about the same topic. This can be seen as an application-oriented variant of textual entailment recognition where: *i)* T and H are in different languages, and *ii)* entailment relations between T and H have to be checked in both directions. Using a combination of lexical, syntactic, and semantic features to train a cross-lingual textual entailment system, we report promising results on different datasets.

1 Introduction

Given two documents about the same topic written in different languages (*e.g.* Wiki pages), content synchronization deals with the problem of automatically detecting and resolving differences in the information they provide, in order to produce aligned, mutually enriched versions. A roadmap towards the solution of this problem has to take into account, among the many sub-tasks, the identification of information in one page that is semantically equivalent, novel, or more informative with respect to the content of the other page. In this paper we set such problem as an application-oriented, cross-lingual variant of the Textual Entailment (TE) recognition task (Dagan and Glickman, 2004). Along this direction, we make two main contributions:

(a) Experiments with multi-directional cross-lingual textual entailment. So far, cross-lingual textual entailment (CLTE) has been only applied to: *i)* available TE datasets (uni-directional relations between monolingual pairs) transformed into their cross-lingual counterpart by translating the hypotheses into other languages (Negri and Mehdad, 2010), and *ii)* machine translation (MT) evaluation datasets (Mehdad et al., 2012). Instead, we experiment with the only corpus representative of the multilingual content synchronization scenario, and the richer inventory of phenomena arising from it (multi-directional entailment relations).

(b) Improvement of current CLTE methods. The CLTE methods proposed so far adopt either a "*pivoting approach*" based on the translation of the two input texts into the same language (Mehdad et al., 2010), or an "*integrated solution*" that exploits bilingual phrase tables to capture lexical relations and contextual information (Mehdad et al., 2011). The promising results achieved with the integrated approach, however, still rely on phrasal matching techniques that disregard relevant semantic aspects of the problem. By filling this gap integrating linguistically motivated features, we propose a novel approach that improves the state-of-the-art in CLTE.

2 CLTE-based content synchronization

CLTE has been proposed by (Mehdad et al., 2010) as an extension of textual entailment which consists of deciding, given a text T and an hypothesis H *in different languages*, if the meaning of H can be inferred from the meaning of T. The adoption of entailment-based techniques to address content synchronization looks promising, as several issues inherent to such task can be formalized as entailment-related prob-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 120–124,
Jeju, Republic of Korea, 8-14 July 2012.

lems. Given two pages (*P1* and *P2*), these issues include identifying, and properly managing:
(1) Text portions in *P1* and *P2* that express the same meaning (bi-directional entailment). In such cases no information has to migrate across *P1* and *P2*, and the two text portions will remain the same;
(2) Text portions in *P1* that are more informative than portions in *P2* (forward entailment). In such cases, the entailing (more informative) portions from *P1* have to be translated and migrated to *P2* in order to replace or complement the entailed (less informative) fragments;
(3) Text portions in *P2* that are more informative than portions in *P1* (backward entailment), and should be translated to replace or complement them;
(4) Text portions in *P1* describing facts that are not present in *P2*, and vice-versa (the "unknown" cases in RTE parlance). In such cases, the novel information from both sides has to be translated and migrated in order to mutually enrich the two pages;
(5) Meaning discrepancies between text portions in the two pages ("contradictions" in RTE parlance).

CLTE has been previously modeled as a phrase matching problem that exploits dictionaries and phrase tables extracted from bilingual parallel corpora to determine the number of word sequences in H that can be mapped to word sequences in T. In this way a *semantic* judgement about entailment is made exclusively on the basis of *lexical* evidence. When only unidirectional entailment relations from T to H have to be determined (RTE-like setting), the full mapping of the hypothesis into the text usually provides enough evidence for a positive entailment judgement. Unfortunately, when dealing with multi-directional entailment, the correlation between the proportion of matching terms and the correct entailment decisions is less strong. In such framework, for instance, the full mapping of the hypothesis into the text is *per se* not sufficient to discriminate between forward entailment and semantic equivalence. To cope with these issues, we explore the contribution of syntactic and semantic features as a complement to lexical ones in a supervised learning framework.

3 Beyond lexical CLTE

In order to enrich the feature space beyond pure lexical match through phrase table entries, our model builds on two additional feature sets, derived from *i)* semantic phrase tables, and *ii)* dependency relations.

Semantic Phrase Table (SPT) matching represents a novel way to leverage the integration of semantics and MT-derived techniques. SPT matching extends CLTE methods based on pure lexical match by means of "generalized" phrase tables annotated with shallow semantic labels. SPTs, with entries in the form "*[LABEL]* $word_1 ... word_n$ *[LABEL]*", are used as a recall-oriented complement to the phrase tables used in MT. A motivation for this augmentation is that semantic tags allow to match tokens that do not occur in the original bilingual parallel corpora used for phrase table extraction. Our hypothesis is that the increase in recall obtained from relaxed matches through semantic tags in place of "out of vocabulary" terms (*e.g.* unseen person names) is an effective way to improve CLTE performance, even at the cost of some loss in precision.

Like lexical phrase tables, SPTs are extracted from parallel corpora. As a first step we annotate the parallel corpora with named-entity taggers for the source and target languages, replacing named entities with general semantic labels chosen from a coarse-grained taxonomy (person, location, organization, date and numeric expression). Then, we combine the sequences of unique labels into one single token of the same label, and we run Giza++ (Och and Ney, 2000) to align the resulting semantically augmented corpora. Finally, we extract the semantic phrase table from the augmented aligned corpora using the Moses toolkit (Koehn et al., 2007). For the matching phase, we first annotate T and H in the same way we labeled our parallel corpora. Then, for each n-gram order (n=1 to 5) we use the SPT to calculate a matching score as the number of n-grams in H that match with phrases in T divided by the number of n-grams in H.[1]

Dependency Relation (DR) matching targets the increase of CLTE precision. Adding syntactic constraints to the matching process, DR features aim to reduce the amount of wrong matches often occurring with bag-of-words methods (both at the lexical level and with recall-oriented SPTs). For instance, the contradiction between "*Yahoo acquired*

[1]When checking for entailment from H to T, the normalization is carried out dividing by the number of n-grams in T.

Overture" and "*Overture compró Yahoo*", which is evident when syntax is taken into account, can not be caught by shallow methods. We define a dependency relation as a triple that connects pairs of words through a grammatical relation. DR matching captures similarities between dependency relations, combining the syntactic and lexical level. In a valid match, while the relation has to be the same, the connected words can be either the same, or semantically equivalent terms in the two languages (*e.g.* according to a bilingual dictionary). Given the dependency tree representations of T and H, for each grammatical relation (*r*) we calculate a DR matching score as the number of matching occurrences of *r* in T and H, divided by the number of occurrences of *r* in H. Separate DR matching scores are calculated for each relation *r* appearing both in T and H.

4 Experiments and results

4.1 Content synchronization scenario

In our first experiment we used the English-German portion of the CLTE corpus described in (Negri et al., 2011), consisting of 500 multi-directional entailment pairs which we equally divided into training and test sets. Each pair in the dataset is annotated with "Bidirectional", "Forward", or "Backward" entailment judgements. Although highly relevant for the content synchronization task, "Contradiction" and "Unknown" cases (*i.e.* "NO" entailment in both directions) are not present in the annotation. However, this is the only available dataset suitable to gather insights about the viability of our approach to multi-directional CLTE recognition.[2] We chose the ENG-GER portion of the dataset since for such language pair MT systems performance is often lower, making the adoption of simpler solutions based on pivoting more vulnerable.

To build the English-German phrase tables we combined the Europarl, News Commentary and "denews"[3] parallel corpora. After tokenization, Giza++ and Moses were respectively used to align the corpora and extract a lexical phrase table (PT). Similarly, the semantic phrase table (SPT) has been extracted from the same corpora annotated with the Stanford NE tagger (Faruqui and Padó, 2010; Finkel et al., 2005). Dependency relations (DR) have been extracted running the Stanford parser (Rafferty and Manning, 2008; De Marneffe et al., 2006). The dictionary created during the alignment of the parallel corpora provided the lexical knowledge to perform matches when the connected words are different, but semantically equivalent in the two languages. To combine and weight features at different levels we used SVMlight (Joachims, 1999) with default parameters.

In order to experiment under testing conditions of increasing complexity, we set the CLTE problem both as a two-way and as a three-way classification task. Two-way classification casts multi-directional entailment as a unidirectional problem, where each pair is analyzed checking for entailment both from left to right and from right to left. In this condition, each original test example is correctly classified if both pairs originated from it are correctly judged ("YES-YES" for bidirectional, "YES-NO" for forward, and "NO-YES" for backward entailment). Two-way classification represents an intuitive solution to capture multidirectional entailment relations but, at the same time, a suboptimal approach in terms of efficiency since two checks are performed for each pair. Three-way classification is more efficient, but at the same time more challenging due to the higher difficulty of multiclass learning, especially with small datasets.

Results are compared with two pivoting approaches, checking for entailment between the original English texts and the translated German hypotheses.[4] The first (Pivot-EDITS), uses an optimized distance-based model implemented in the open source RTE system EDITS (Kouylekov and Negri, 2010; Kouylekov et al., 2011). The second (Pivot-PPT) exploits paraphrase tables for phrase matching, and represents the best monolingual model presented in (Mehdad et al., 2011). Table 1 demonstrates the success of our results in proving the two main claims of this paper. **(a)** In both settings all the feature sets used outperform the approaches taken as terms of comparison. The 61.6% accuracy achieved in the most challenging setting

[2]Recently, a new dataset including "Unknown" pairs has been used in the "*Cross-Lingual Textual Entailment for Content Synchronization*" task at SemEval-2012 (Negri et al., 2012).

[3]http://homepages.inf.ed.ac.uk/pkoehn/

[4]Using Google Translate.

	PT	PT+DR	PT+SPT	PT+SPT+DR	Pivot-EDITS	Pivot-PPT
Cont. Synch. (2-way)	57.8	58.6	62.4	**63.3**	27.4	57.0
Cont. Synch. (3-way)	57.4	57.8	58.7	**61.6**	25.3	56.1
					RTE-3 AVG	Pivot PPT
RTE3-derived	62.6	63.6	63.5	**64.5**	62.4	63.5

Table 1: CLTE accuracy results over content synchronization and RTE3-derived datasets.

(3-way) demonstrates the effectiveness of our approach to capture meaning equivalence and information disparity in cross-lingual texts.
(b) In both settings the combination of lexical, syntactic and semantic features (PT+SPT+DR) significantly improves[5] the state-of-the-art CLTE model (PT). Such improvement is motivated by the joint contribution of SPTs (matching more and longer n-grams, with a consequent recall improvement), and DR matching (adding constraints, with a consequent gain in precision). However, the performance increase brought by DR features over PT is minimal. This might be due to the fact that both PT and DR features are precision-oriented, and their effectiveness becomes evident only in combination with recall-oriented features (SPT).

Cross-lingual models also significantly outperform pivoting methods. This suggests that the noise introduced by incorrect translations makes the pivoting approach less attractive in comparison with the more robust cross-lingual models.

4.2 RTE-like CLTE scenario

Our second experiment aims at verifying the effectiveness of the improved model over RTE-derived CLTE data. To this aim, we compare the results obtained by the new CLTE model with those reported in (Mehdad et al., 2011), calculated over an English-Spanish entailment corpus derived from the RTE-3 dataset (Negri and Mehdad, 2010).

In order to build the English-Spanish lexical phrase table (PT), we used the Europarl, News Commentary and United Nations parallel corpora. The semantic phrase table (SPT) was extracted from the same corpora annotated with FreeLing (Carreras et al., 2004). Dependency relations (DR) have been extracted parsing English texts and Spanish hypotheses with DepPattern (Gamallo and Gonzalez, 2011).

[5] $p < 0.05$, calculated using the approximate randomization test implemented in (Padó, 2006).

Accuracy results have been calculated over 800 test pairs of the CLTE corpus, after training the SVM binary classifier over the 800 development pairs. Our new features have been compared with: *i)* the state-of-the-art CLTE model (PT), *ii)* the best monolingual model (Pivot-PPT) presented in (Mehdad et al., 2011), and *iii)* the average result achieved by participants in the monolingual English RTE-3 evaluation campaign (RTE-3 AVG). As shown in Table 1, the combined feature set (PT+SPT+DR) significantly[5] outperforms the lexical model (64.5% vs 62.6%), while SPT and DR features separately added to PT (PT+SPT, and PT+DR) lead to marginal improvements over the results achieved by the PT model alone (about 1%). This confirms the conclusions drawn from the previous experiment, that precision-oriented and recall-oriented features lead to a larger improvement when they are used in combination.

5 Conclusion

We addressed the identification of semantic equivalence and information disparity in two documents about the same topic, written in different languages. This is a core aspect of the multilingual content synchronization task, which represents a challenging application scenario for a variety of NLP technologies, and a shared research framework for the integration of semantics and MT technology. Casting the problem as a CLTE task, we extended previous lexical models with syntactic and semantic features. Our results in different cross-lingual settings prove the feasibility of the approach, with significant state-of-the-art improvements also on RTE-derived data.

Acknowledgments

This work has been partially supported by the EU-funded project CoSyne (FP7-ICT-4-248531).

References

X. Carreras, I. Chao, L. Padró, and M. Padró. 2004. FreeLing: An Open-Source Suite of Language Analyzers. In *Proceedings of the 4th Language Resources and Evaluation Conference (LREC 2004)*, volume 4.

I. Dagan and O. Glickman. 2004. Probabilistic Textual Entailment: Generic Applied Modeling of Language Variability. In *Proceedings of the PASCAL Workshop of Learning Methods for Text Understanding and Mining*.

M.C. De Marneffe, B. MacCartney, and C.D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the 5th Language Resources and Evaluation Conference (LREC 2006)*, volume 6, pages 449–454.

M. Faruqui and S. Padó. 2010. Training and Evaluating a German Named Entity Recognizer with Semantic Generalization. In *Proceedings of the 10th Conference on Natural Language Processing (KONVENS 2010)*, Saarbrücken, Germany.

J.R. Finkel, T. Grenager, and C. 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 2005)*.

P. Gamallo and I. Gonzalez. 2011. A grammatical formalism based on patterns of part of speech tags. *International Journal of Corpus Linguistics*, 16(1):45–71.

T. Joachims. 1999. Advances in kernel methods. chapter Making large-scale support vector machine learning practical, pages 169–184. MIT Press, Cambridge, MA, USA.

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 on Association for Computational Linguistics, Demonstration Session (ACL 2007)*.

M. Kouylekov and M. Negri. 2010. An Open-Source Package for Recognizing Textual Entailment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, system demonstrations (ACL 2010)*.

M. Kouylekov, Y. Mehdad, and M. Negri. 2011. Is it Worth Submitting this Run? Assess your RTE System with a Good Sparring Partner. *Proceedings of the EMNLP TextInfer 2011 Workshop on Textual Entailment*.

Y. Mehdad, M. Negri, and M. Federico. 2010. Towards Cross-Lingual Textual Entailment. In *Proceedings of the 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*.

Y. Mehdad, M. Negri, and M. Federico. 2011. Using Bilingual Parallel Corpora for Cross-Lingual Textual Entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*.

Y. Mehdad, M. Negri, and M. Federico. 2012. Match without a Referee: Evaluating MT Adequacy without Reference Translations. In *Proceedings of the Machine Translation Workshop (WMT2012)*.

M. Negri and Y. Mehdad. 2010. Creating a Bi-lingual Entailment Corpus through Translations with Mechanical Turk: $100 for a 10-day Rush. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazonś Mechanical Turk*.

M. Negri, L. Bentivogli, Y. Mehdad, D. Giampiccolo, and A. Marchetti. 2011. Divide and Conquer: Crowdsourcing the Creation of Cross-Lingual Textual Entailment Corpora. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.

M. Negri, A. Marchetti, Y. Mehdad, L. Bentivogli, and D. Giampiccolo. 2012. Semeval-2012 Task 8: Cross-lingual Textual Entailment for Content Synchronization. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012)*.

F.J. Och and H. Ney. 2000. Improved Statistical Alignment Models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL 2000)*.

S. Padó, 2006. *User's guide to* `sigf`*: Significance testing by approximate randomisation*.

A.N. Rafferty and C.D. Manning. 2008. Parsing Three German Treebanks: Lexicalized and Unlexicalized Baselines. In *In Proceedings of the ACL 2008 Workshop on Parsing German*.

Cross-lingual Parse Disambiguation based on Semantic Correspondence

Lea Frermann
Department of Computational Linguistics
Saarland University
frermann@coli.uni-saarland.de

Francis Bond
Linguistics and Multilingual Studies
Nanyang Technological University
bond@ieee.org

Abstract

We present a system for cross-lingual parse disambiguation, exploiting the assumption that the meaning of a sentence remains unchanged during translation and the fact that different languages have different ambiguities. We simultaneously reduce ambiguity in multiple languages in a fully automatic way. Evaluation shows that the system reliably discards dispreferred parses from the raw parser output, which results in a pre-selection that can speed up manual treebanking.

1 Introduction

Treebanks, sets of parsed sentences annotated with a sytactic structure, are an important resource in NLP. The manual construction of treebanks, where a human annotator selects a gold parse from all parses returned by a parser, is a tedious and error prone process. We present a system for simultaneous and accurate partial parse disambiguation of multiple languages. Using the pre-selected set of parses returned by the system, the treebanking process for multiple languages can be sped up.

The system operates on an aligned parallel corpus. The languages of the parallel corpus are considered as mutual semantic tags: As the meaning of a sentence stays constant during translation, we are able to resolve ambiguities which exist in only one of the langauges by only accepting those interpretations which are licensed by the other language.

In particular, we select one language as the target language, translate the other language's semantics for every parse into the target language and thus align maximally similar semantic representations. The parses with the most overlapping semantics are selected as preferred parses.

As an example consider the English sentence *They closed the shop at five*, which has the following two interpretations due to PP attachment ambiguity:[1]

(1) "At five, they closed the shop"

`close(they, shop); at(close, 5)`

(2) "The shop at five was closed by them"

`close(they, shop); at(shop, 5)`

The Japanese translation is also ambiguous, but in a completely different way: it has the possibility of a zero pronoun (we show the translated semantics).

(3) 彼 ら は 5 時 に 店 を 閉め た
kare ra wa 5 ji ni mise wo shime ta
he PL TOP 5 hour at shop ACC close PAST
"At 5 o'clock, they closed the shop."

`close(they, shop); at(close, 5)`

(4) "At 5 o'clock, as for them, someone closed the shop."

`close(`ϕ`, shop); at(close, 5)`
`topic(they,close)`

We show the semantic representation of the ambiguity with each sentence. Both languages are disambiguated by the other language as only the English interpretation (1) is supported in Japanese, and only the Japanese interpretation (3) leads to a grammatical English sentence.

2 Related Work

There is no group using exactly the same approach as ours: automated parallel parse disambiguation on the basis of semantic analyses. Zhechev and

[1] In fact it has four, as *they* can be either plural or the androgynous singular, this is also disambiguated by the Japanese.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 125–129,
Jeju, Republic of Korea, 8-14 July 2012.

Way (2008) automatically generate parallel treebanks for training of statistical machine translation (SMT) systems through sub-tree alignment. We do not aim to carry out the complete treebanking process, but to optimize speed and precision of manual creation of high-quality treebanks.

Wu (1997) and others have tried to simultaneously learn grammars from bilingual texts. Burkett and Klein (2008) induce node-alignments of syntactic trees with a log-linear model, in order to guide bilingual parsing. Chen et al. (2011) translate an existing treebank using an SMT system and then project parse results from the treebank to the other language. This results in a very noisy treebank, that they then clean. These approaches align at the syntactic level (using CFGs and dependencies respectively).

In contrast to the above approaches, we assume the existence of grammars and use a semantic representation as the appropriate level for cross-lingual processing. We compare semantic sub-structures, as those are more straightforwardly comparable across different languages. As a consequence, our system is applicable to any combination of languages. The input is plain parallel text, neither side needs to be treebanked.

3 Materials and Methods

We use grammars within the grammatical framework of head-driven phrase-structure grammar (HPSG Pollard and Sag (1994)), with the semantic representation of minimal recursion semantics (MRS; Copestake et al. (2005)). We use two large-scale HPSG grammars and a Japanese-English machine translation system, all of which were developed in the DELPH-IN framework:[2] The English Resource Grammar (ERG; Flickinger (2000)) is used for English parsing, and Jacy (Bender and Siegel, 2004) for parsing Japanese. For Japanese to English translation we use Jaen, a semantic-transfer based machine translation system (Bond et al., 2011).

3.1 Semantic Interface and Alignment

For the alignment, we convert the MRS structures into simplified elementary dependency graphs (EDGs), which abstract away information about grammatical properties of relations and scopal information. Preliminary experiments showed that the former kind of information did not contribute to disambiguation performance, as number is typically underspecified in Japanese. As we only consider local information in the alignment, scopal information can be ignored as well. An example EDG is displayed in Figure 1.

```
x4:pronoun_q[]
e2:_close_v_c[ARG1 x4:pron, ARG2 x9:_shop_n_of]
x9:_the_q[]
e8:_at_p_temp[ARG1 e2, ARG2 x16:_num_hour(5)]
x16:_def_implicit_q[]
```

Figure 1: EDG for *They closed the shop at five*.

An EDG consists of a bag of elementary predicates (EPs) which are themselves composed of relations. Each line in Figure 1 corresponds to one EP. Relations are the elementary building blocks of the EDG, and loosely correspond to words of the surface string. EPs consist either of atomic relations (corresponding to quantifiers), or a predicate-argument structure which is composed of several relations. During alignment, we only consider non-atomic EPs, as quantifiers should be considered as grammatical properties of (lexical) relations, which we chose to ignore.

Given the EDG representations of the translated Japanese sentence, and the original target language EDGs, we can straightforwardly align by matching substructures of different granularity.

Currently, we align at the predicate level. We are experimenting with aligning further dependency relation based tuples, which would allow us to resolve more structural ambiguities.

3.2 The Disambiguation System

Ambiguity in the analyses for both languages is reduced on the basis of the semantic analyses returned for each sentence-pair, and a reduced set of preferred analyses is returned for both languages. For each sentence-pair, we (1) parse the English and the Japanese sentence (MRS_E and MRS_J) (2) transfer the Japanese MRS analyses to English MRSs (MRS_{JE}) (3) convert the top 11 translated MRSs

[2] http://www.delph-in.net/

and the original English MRSs to EDGs[3] (EDG_E and EDG_{JE}) (4) align every possible E and JE EDG combination and determine the set of best aligning analyses (5) from those, create language specific sets of preferred parses.

We are comparing semantic representations of the **same** language, the English text from the bilingual corpus and the English machine translation of the Japanese text. In order to increase robustness of our alignment system we not only consider complete translations, but also accept partially translated MRSs in case no complete translation could be produced. This step significantly increases the recall, while the partial MRSs proved to be informative enough for parse disambiguation.

4 Evaluation and Results

We evaluate our model on the task of parse disambiguation. We use full sentence match as evaluation metric, a challenging target.

The Tanaka corpus is used for training and testing (Tanaka, 2001). It is an open corpus of Japanese-English sentence pairs. We use version (2008-11) which contains 147,190 sentence pairs. We hold out 4,500 sentence pairs each for development and test.

For each sentence, we compare the number of theoretically possible alignments with the number of preferred alignments returned by our system. On average, ambiguity is reduced down to 30%. For English 3.76 and for Japanese 3.87 parses out of (at most) 11 analyses remain in the partially disambiguated list: both languages benefit equally from the disambiguation.

We evaluate disambiguation accuracy by counting the number of times the gold parse was present in the partially disambiguated set (full sentence match). Table 1 shows the alignment accuracy results.

The correct parse is included in the reduced set in 80% of the cases for Japanese, and for 82% of the cases in English. We match atomic relations when aligning the semantic structures, which is a very generic method applicable to the vast majority of sentence pairs. This leads to a recall score of 99%, and an F-Score of 89.7% and 88.7% for English and Japanese, respectively.

[3]These are ranked with a model trained on a hand-treebanked set. The cutoff was determined empirically: For both languages the gold parse is included in the top 11 parses in more than 97% of the cases.

	English		Japanese	
	Prec	**F**	**Prec**	**F**
Included	0.820	0.897	0.804	0.887
First Rank	0.659	0.791	0.676	0.803
MRR	0.713	0.829	0.725	0.837

Table 1: Accuracy and F-scores for disambiguation performance of our system. Recall was 99% in every case. 'Included': inclusion of the gold parse in the reduced set of parses or not. 'First Rank': ranking of the preferred parse as top in the reduced list. 'MRR': mean reciprocal rank of the gold parse in the list.

The reduced list of parser analyses can be further ranked by the parse ranking model which is included in the parsers of the respective languages (the same models with which we determined the top 11 analyses). Given this ranking, we can evaluate how often the preferred parse is ranked top in our partially disambiguated list; results are shown in the two bottom lines of Table 1.

A ranked list of possible preferred parses whose top rank corresponds with a high probability to the gold parse should further speed up the manual treebanking process.

Performance in the context of the whole pipeline

The performance of parsers and MT system strongly influences the end-to-end results of the presented system. In the results given above, this influence is ignored. We lose around 29% of our data because no parse could be produced in one or both languages, or no translation could be produced. and a further 5% of the sentences did not have the gold parse in the original set of analyses (*before* alignment): our system could not possibly select the correct parse in those cases.

5 Discussion

Our system builds on the output of two parsers and a machine translation system. We reduce ambiguity for all sentence pairs where a parse could be created for both languages, and for which there was at least a partial translation. For these sentences, the cross-lingual alignment component achieves a recall of above 99%, such that we do not lose any addi-

tional data. The parsers and the MT system include a parse ranking system trained on human gold annotations. We use these models in parsing and translation to select the top 11 analyses. Our system thus depends on a range of existing technologies. However, these technologies are available for a range of languages, and we use them for efficient extension of linguistic resources.

The effectiveness of cross-lingual parse disambiguation on the basis of semantic alignment highly depends on the languages of choice. Given that we exploit the differences between languages, pairs of less related languages should lead to better disambiguation performance. Furthermore, disambiguating with more than two languages should improve performance. Some ambiguities may be shared between languages. [4]

One weakness when considering the disambiguated sentences as training for a parse ranking model is that the translation fails on similar kinds of sentences, so there are some phenomena which we get no examples of — the automatically trained treebank does not have a uniform coverage of phenomena. Our models may not discriminate some phenomena at all.

Our system provides large amounts of automatically annotated data at the only cost of CPU time: so far we have disambiguated 25,000 sentences: 10 times more than the existing hand annotated gold data. Using the parser output for speeding up manual treebanking is most effective if the gold parse is reliably included in the reduced set of parses. Increasing precision by accepting more than only the most overlapping parses may lead to more effective manual treebanking.

The alignment method we propose does not make any language-specific assumptions, nor is it limited to align two languages only. The algorithm is very flexible, and allows for straightforward exploration of different numbers and combinations of languages.

6 Conclusion and Future Work

Translating a sentence into a different language changes its surface form, but not its meaning. In parallel corpora, one language can be viewed as a semantic tag of the other language and vice versa, which allows for disambiguation of phenomena which are ambiguous in only one of the languages.

We use the above observations for cross-lingual parse disambiguation. We experimented with the language pair of English and Japanese, and were able to accurately reduce ambiguity in parser analyses simultaneously for both languages to 30% of the starting ambiguity. The remaining parses can be used as a pre-selection to speed up the manual treebanking process.

We started working on an extrinsic evaluation of the presented system by training a discriminative parse ranking model on the output of our alignment process. Augmenting the Gold training data with our data improves the model. Our next step will be to evaluate the system as part of the treebanking process, and optimize the parameters such as disambiguation precision vs. amount of disambiguation.

As no language-specific assumptions are hard coded in our disambiguation system, it would be very interesting to apply the system to different language pairs as well as groups of more than two languages. Using a group of languages for disambiguation will likely lead to increased and more accurate disambiguation, as more constraints are imposed on the data.

Probably the most important goal for future work is improving the recall achieved in the complete disambiguation pipeline. Many sentence-pairs cannot be disambiguated because either no parse can be generated for one or both languages, or no (partial) translation can be produced. Following the idea of partial translations, partial parses may be a valid backoff. For purposes of cross-lingual alignment, partial structures may contribute enough information for disambiguation. There has been work regarding partial parsing in the HPSG community (Zhang and Kordoni, 2008), which we would like to explore. There is also current work on learning more types and instances of transfer rules (Haugereid and Bond, 2011).

Finally, we would like to investigate more alignment methods, such as dependency relation based alignment which we started experimenting with, or EDM-based metrics as presented in (Dridan and Oepen, 2011).

[4] For example the PP attachment ambiguity in *John said that he went on Tuesday* where either the saying or the going could have happened on Tuesday holds in both English and Japanese.

Acknowledgments

This research was supported in part by the Erasmus Mundus Action 2 program MULTI of the European Union, grant agreement number 2009-5259-5 and the the joint JSPS/NTU grant on *Revealing Meaning Using Multiple Languages*. We would like to thank Takayuki Kuribayashi and Dan Flickinger for their help with the treebanking.

References

Emily M. Bender and Melanie Siegel. 2004. Implementing the syntax of Japanese numeral classifiers. In *Proceedings of the IJC-NLP-2004*.

Francis Bond, Stephan Oepen, Eric Nichols, Dan Flickinger, Erik Velldal, and Petter Haugereid. 2011. Deep open-source machine translation. *Machine Translation*, 25(2):87–105.

David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *Proceedings of EMNLP, 2008*.

Wenliang Chen, Jun'ichi Kazama, Min Zhang, Yoshimasa Tsuruoka, Yujie Zhang, Yiou Wang, Kentaro Torisawa, and Haizhou Li. 2011. SMT helps bitext dependency parsing. In *Conference on Empirical Methods in Natural Language Processing (EMNLP2011)*, pages 73–83. Edinburgh.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A. Sag. 2005. Minimal recursion semantics – an introduction. *Research on Language and Computation*, 3:281–332.

Rebecca Dridan and Stephan Oepen. 2011. Parser evaluation using elementary dependency matching. In *Proceedings of IWPT*.

Dan Flickinger. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering*, 6(1):15–28. (Special Issue on Efficient Processing with HPSG).

Petter Haugereid and Francis Bond. 2011. Extracting transfer rules for multiword expressions from parallel corpora. In *Proceedings of the Workshop on Multiword Expressions: from Parsing and Generation to the Real World*.

Carl Pollard and Ivan A. Sag. 1994. *Head Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Yasuhito Tanaka. 2001. Compilation of a multilingual parallel corpus. In *Proceedings of PACLING 2001*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Yi Zhang and Valia Kordoni. 2008. Robust parsing with a large HPSG grammar. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*.

Ventsislav Zhechev and Andy Way. 2008. Automatic generation of parallel treebanks. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*.

Learning to Find Translations and Transliterations on the Web

Joseph Z. Chang
Department of Computer Science,
National Tsing Hua University
101, Kuangfu Road,
Hsinchu, 300, Taiwan
joseph.nthu.tw@gmail.com

Jason S. Chang
Department of Computer Science,
National Tsing Hua University
101, Kuangfu Road,
Hsinchu, 300, Taiwan
jschang@cs.nthu.edu.tw

Jyh-Shing Roger Jang
Department of Computer Science,
National Tsing Hua University
101, Kuangfu Road,
Hsinchu, 300, Taiwan
jang@cs.nthu.edu.tw

Abstract

In this paper, we present a new method for learning to finding translations and transliterations on the Web for a given term. The approach involves using a small set of terms and translations to obtain mixed-code snippets from a search engine, and automatically annotating the snippets with tags and features for training a conditional random field model. At run-time, the model is used to extracting translation candidates for a given term. Preliminary experiments and evaluation show our method cleanly combining various features, resulting in a system that outperforms previous work.

1 Introduction

The phrase translation problem is critical to machine translation, cross-lingual information retrieval, and multilingual terminology (Bian and Chen 2000, Kupiec 1993). Such systems typically use a parallel corpus. However, the out of vocabulary problem (OOV) is hard to overcome even with a very large training corpus due to the Zipf nature of word distribution, and ever growing new terminology and named entities. Luckily, there are an abundant of webpages consisting mixed-code text, typically written in one language but interspersed with some sentential or phrasal translations in another language. By retrieving and identifying such translation counterparts on the Web, we can cope with the OOV problem.

Consider the technical term *named-entity recognition*. The best places to find the Chinese translations for named-entity recognition are probably not some parallel corpus or dictionary, but rather mixed-code webpages. The following example is a snippet returned by the Bing search engine for the query, *named entity recognition*:

> ... 語言處理技術，如自然語言剖析 (Natural Language Parsing)、問題分類 (Question Classification)、專名辨識 (Named Entity Recognition)等等 ...

This snippet contains three technical terms in Chinese (i.e., 自然語言剖析 *zhiran yuyan poxi*, 問題分類 *wenti fenlei*, 專名辨識 *zhuanming bianshi*), followed by source terms in brackets (respectively, *Natural Language Parsing*, *Question Classification*, and *Named Entity Recognition*). Quoh (2006) points out that submitting the source term and partial translation to a search engine is a good strategy used by many translators.

Unfortunately, the user still has to sift through snippets to find the translations. For a given English term, such translations can be extracted by casting the problem as a sequence labeling task for classifying the Chinese characters in the snippets as either translation or non-translation. Previous work has pointed out that such translations usually exhibit characteristics related to word translation, word transliteration, surface patterns, and proximity to the occurrences of the original phrase (Nagata et. al 2001 and Wu et. al 2005).

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 130–134,
Jeju, Republic of Korea, 8-14 July 2012.

Thus, we also associate features to each Chinese token (characters or words) to reflect the likelihood of the token being part of the translation. We describe how to train a CRF model for identifying translations in more details in Section 3.

At run-time, the system accepts a given phrase (e.g., *named-entity recognition*), and then query a search engine for webpages in the target language (e.g., Chinese) using the advance search function. Subsequently, we retrieve mixed-code snippets and identify the translations of the given term. The system can potentially be used to assist translators to find the most common translation for a given term, or to supplement a bilingual terminology bank (e.g., adding multilingual titles to existing Wikipedia); alternatively, they can be used as additional training data for a machine translation system, as described in Lin et al. (2008).

	Token	TR	TL	Distance	Label
	第	0	0	14	O
	62	0	0	13	O
62th	屆	0	0	12	O
	艾	**3**	0	11	**B**
Emmy	美	**3**	0	10	**I**
Award	獎	0	**5**	9	**I**
	頒	0	0	8	O
awarding	獎	0	0	7	O
	典	0	0	6	O
ceremony	禮	0	0	5	O
	》	0	0	4	O
	(	0	0	3	O
	the	0	0	2	O
	62th	0	0	1	O
	Emmy	0	0	0	E
	Award	0	0	0	E
	)	0	0	-1	O

Figure 1. Example training data.

2 Related Work

Phrase translation and transliteration is important for cross-language tasks. For example, Knight and Graehl (1998) describe and evaluate a multi-stage machine translation method for back transliterating English names into Japanese, while Bian and Chen (2000) describe cross-language information access to multilingual collections on the Internet.

Recently, researchers have begun to exploit mixed code webpages for word and phrase translation. Nagata et al. (2001) present a system for finding English translations for a given Japanese technical term using Japanese-English snippets returned by a search engine. Kwok et al. (2005) focus on named entity transliteration and implemented a cross-language name finder. Wu et al. (2005) proposed a method to learn surface patterns to find translations in mixed code snippets.

Some researchers exploited the hyperlinks in Webpage to find translations. Lu, et al. (2004) propose a method for mining translations of web queries from anchor texts. Cheng, et al (2004) propose a similar method for translating unknown queries with web corpora for cross-language information retrieval. Gravano (2006) also propose similar methods using anchor texts.

In a study more closely related to our work, Lin et al. (2008) proposed a method that performs word alignment between translations and phrases within parentheses in crawled webpages. They use heuristics to align words and translations, while we use a learning based approach to find translations.

In contrast to previous work described above, we exploit surface patterns differently as a soft constraint, while requiring minimal human intervention to prepare the training data.

3 Method

To find translations for a given term on the Web, a promising approach is automatically learning to extract phrasal translations or transliterations of phrase based on machine learning, or more specifically the conditional random fields (CRF) model.

We focus on the issue of finding translations in mixed code snippets returned by a search engine. The translations are identified, tallied, ranked, and returned as the output of the system.

3.1 Preparing Data for CRF Classifier

We make use a small set of term and translation pairs as seed data to retrieve and annotate mixed-code snippets from a search engine. Features are generated based on other external knowledge sources as will be described in Section 3.1.2 and 3.1.3. An example data generated with given term *Emmy Award* with features and translation/non-translation labels is shown in Figure 1 using the common **BIO** notation.

3.1.1 *Retrieving and tagging snippets*. We use a list of randomly selected source and target terms as seed data (e.g., Wikipedia English titles and their

Chinese counterpart using the *language links*). We use the English terms (e.g., *Emmy Awards*) to query a search engine with the target webpage language set to the target language (e.g., Chinese), biasing the search engine to return Chinese webpages interspersed with some English phrases. We then automatically label each Chinese character of the returned snippets, with **B**, **I**, **O** indicating respectively ***beginning***, ***inside***, and ***outside*** of translations. In Figure 1, the translation 艾美獎 (*ai mei jiang*) are labeled as **B I I**, while all other Chinese characters are labeled as **O**. An additional tag of ***E*** is used to indicate the occurrences of the given term (e.g., *Emmy Awards* in Figure 1).

3.1.2 *Generating translation feature*. We generate translation features using external bilingual resources. The φ^2 score proposed by Gale and Church (1991) is used to measure the correlations between English and Chinese tokens:

$$\varphi^2 = \frac{[P(e,f)P(\bar{e},\bar{f}) - P(\bar{e},f)P(e,\bar{f})]^2}{P(e)P(f)P(\bar{e})P(\bar{f})}$$

where e is an English word and f is a Chinese character. The scores are calculated by counting co-occurrence of Chinese characters and English words in bilingual dictionaries or termbanks, where $P(e, f)$ represents the probability of the co-occurrence of English word *e* and Chinese character *f*, and $P(e, \bar{f})$ represents the probability the co-occurrence of *e* and any Chinese characters excluding *f*.

We used the publicly available English-Chinese Bilingual WordNet and NICT terminology bank to generate translation features in our implementation. The bilingual WordNet has 99,642 synset entries, with a total of some 270,000 translation pairs, mainly common nouns. The NICT database has over 1.1 million bilingual terms in 72 categories, covering a wide variety of different fields.

3.1.3 *Generating transliteration feature*. Since many terms are transliterated, it is important to include transliteration feature. We first use a list of name transliterated pairs, then use Expectation-Maximization (EM) algorithm to align English syllables Romanized Chinese characters. Finally, we use the alignment information to generate transliteration feature for a Chinese token with respect to English words in the query.

We extract person or location entries in Wikipedia as name transliterated pairs to generate transliteration features in our implementation. This can be achieved by examining the Wikipedia categories for each entry. A total of some 15,000 bilingual names of persons and 24,000 bilingual place names were obtained and forced aligned to obtain transliteration relationships.

3.1.4 *Generating distance feature*. In the final stage of preparing training data, we add the distance, i.e. number of words, between a Chinese token feature and the English term in question, aimed at exploiting the fact that translations tend to occur near the source term, as noted in Nagata et al. (2001) and Wu et al. (2005).

Finally, we use the data labeled with translation tags and three kinds feature values to train a CRF model.

3.2 Run-Time Translation Extraction

With the trained CRF model, we then attempt to find translations for a given phrase. The system begins by submitting the given phrase as query to a search engine to retrieve snippets, and generate features for each tokens in the same way as done in the training phase. We then use the trained model to tag the snippets, and extract translation candidates by identifying consecutive Chinese tokens labeled as **B** and **I**.

Finally, we compute the frequency of all the candidates identified in all snippets, and output the one with the highest frequency.

4 Experiments and Evaluation

We extracted the Wikipedia titles of English and Chinese articles connected through language links for training and testing. We obtained a total of 155,310 article pairs, from which we then randomly selected 13,150 and 2,181 titles as seeds to obtain the training and test data. Since we are using Wikipedia bilingual titles as the gold standard, we exclude any snippets from the *wikipedia.org* domain, so that we are not using Wikipedia article content in both training and testing stage. The test set contains 745,734 snippets or 9,158,141 tokens (Chinese character or English word). The reference answer appeared a total of 48,938 times or 180,932 tokens (2%), and an average of 22.4 redundant answer instances per input.

System	Coverage	Exact match	Top5 exact match
Full (En-Ch)	**80.4%**	**43.0%**	**56.4%**
-TL	83.9%	27.5%	40.2%
-TR	81.2%	37.4%	50.3%
-TL-TR	83.2%	21.1%	32.8%
LIN En-Ch	59.6%	27.9%	not reported
LIN Ch-En	70.8%	36.4%	not reported
LCD (En-Ch)	10.8%	4.8%	N/A
NICT (En-Ch)	24.2%	32.1%	N/A

Table 1. Automatic evaluation results of 8 experiments: (1) Full system (2-4) -TL, -TR, -TL-TR : Full system deprecating TL, TR, and TL+TL features (5,6) LIN En-Ch and En-Ch : the results in Lin et al. (2008) (6) LDC: LDC E-C dictionary (7) NICT : NICT term bank.

English Wiki	Chinese Wiki	Extracted	Ev.
Pope Celestine IV	塞萊斯廷四世	切萊斯廷四世	A
Fujian	福建省	福建	A
Waste	垃圾	廢物	A
Collateral	落日殺神	抵押	B
Ludwig Erhard	路德維希·艾哈德	艾哈德	P
Osman I	奧斯曼一世	奧斯曼	P
Bubble sort	冒泡排序	排序	P
The Love Suicides at Sonezaki	曾根崎情死	夏目漱石	E
Ammonium	**銨**	過硫酸銨	E

Table 2. Cases failing the exact match test.

Result	Count	Percentage
A+B: correct	53	55.8%
P: partially corr.	30	31.6%
E: incorrect	8	8.4%
N: no results	4	4.2%
total	95	100%

Table 3. Manual evaluation of unlink titles.

To compare our method with previous work, we used a similar evaluation procedure as described in Lin et al. (2008). We ran the system and produced the translations for these 2,181 test data, and automatically evaluate the results using the metrics of ***coverage***, i.e. when system was able to produce translation candidates, and ***exact match precision***.

This precision rate is an under-estimations, since a term may have many alternative translations that does not match exactly with one single reference translation. To give a more accurate estimate of real precision, we resorted to manual evaluation on a small part of the 2,181 English phrases and a small set of English Wikipedia titles without a Chinese language link.

4.1 Automatic Evaluation

In this section, we describe the evaluation based on English-Chinese titles extracted from Wikipedia as the gold standard. Our system produce the top-1 translations by ranking candidates by frequency and output the most frequent translations. Table 1 shows the results we have obtained as compared to the results of Lin et al. (2008).

Table 1 shows the evaluation results of 8 experiments. The results indicate that using external knowledge to generate feature improves system performance significantly. By adding translation feature (TL) or transliteration feature (TR) to the system with no external knowledge features (-TL-TR) improves exact match precision by about 6% and 16% respectively. Because many Wikipedia titles are named entities, transliteration feature is the most important. Overall, the system with full features perform the best, finding reasonably correct translations for 8 out of 10 phrases.

4.2 Manual Evaluation

Evaluation based on exact match against a single reference answer leads to under-estimation, because an English phrase is often translated into several Chinese counterparts. Therefore, we asked a human judge to examine and mark the outputs of our full system. The judge was instructed to mark each output as **A**: correct translation alternative, **B**: correct translation but with a difference sense from the reference, **P**: partially correct translation, and **E**: incorrect translation.

Table 2 shows some translations generated by the full system that does not match the single reference translation. Half of the translations are correct translations (**A** and **B**), while a third are partially correct translation (**P**). Notice that it is a common practice to translate only the surname of a foreign person. Therefore, some partial translations may still be considered as correct (**B**).

To Evaluate titles without a language link, we sampled a list of 95 terms from the unlinked portion of Wikipedia using the criteria: (1) with a frequency count of over 2,000 in Google Web 1T. (2) containing at least three English words. (3) not a proper name. Table 3 shows the evaluation

results. Interestingly, our system provides correct translations for over 50% of the cases, and at least partially correct almost 90% of the cases.

5 Conclusion and Future work

We have presented a new method for finding translations on the Web for a given term. In our approach, we use a small set of terms and translations as seeds to obtain and to tag mixed-code snippets returned by a search engine, in order to train a CRF model for sequence labels. This CRF model is then used to tag the returned snippets for a given query term to extraction translation candidates, which are then ranked and returned as output. Preliminary experiments and evaluations show our learning-based method cleanly combining various features, producing quality translations and transliterations.

Many avenues exist for future research and improvement. For example, existing query expansion methods could be implemented to retrieve more webpages containing translations. Additionally, an interesting direction to explore is to identify phrase types and train type-specific CRF model. In addition, natural language processing techniques such as word stemming and word lemmatization could be attempted.

References

G. W. Bian, H. H. Chen. Cross-language information access to multilingual collections on the internet. 2000. Journal of American Society for Information Science & Technology (JASIST), Special Issue on Digital Libraries, 51(3), pp.281-296, 2000.

Y. Cao and H. Li. Base Noun Phrase Translation Using Web Data and the EM Algorithm. 2002. In *Proceedings* of the 19th International Conference on Computational Linguistics (COLING'02), pp.127-133, 2002.

P. J. Cheng, J. W. Teng, R. C. Chen, J. H. Wang, W. H. Lu, and L. F. Chien. Translating unknown queries with web corpora for cross-language information retrieval. In Proceedings of the 27th ACM International Conference on Research and Development in Information Retrieval, pp.146-153, 2004.

F. Huang, S. Vogel, and A. Waibel. Automatic extraction of named entity translingual equivalence based on multi-feature cost minimization. In *Proceeding of* the 41st ACL, Workshop on Multilingual and Mixed-Language Named Entity Recognition, Sapporo, 2003.

K. Knight, J. Graehl. Machine Transliteration. 1998. Computational Linguistics 24(4), pp.599-612, 1998.

P. Koehn, K. Knight. 2003. Feature-Rich Statistical Translation of Noun Phrases. In *Proceedings of the* 41st Annual Meeting on Association for Computational *Linguistics*, pp. 311-318, 2003.

J. Kupiec. 1993. An Algorithm for Finding Noun Phrase Correspondences in Bilingual Corpora. In *Proceedings* of the 31st Annual Meeting of the Association for Computational Linguistics, pp. 17-22, 1993.

KL Kwok, P Deng, N Dinstl, HL Sun, W Xu, P Peng, and Doyon, J. 2005. CHINET: a Chinese name finder system for document triage. In *Proceedings of 2005*

D. Lin, S. Zhao, B.V. Durme, and M. Paşca. 2008. Mining Parenthetical Translation from the Web by Word Alignment, In Proceedings of ACL 2008, pp. 994-1002, 2008.

Y. Li, G. Grefenstette. 2005. Translating Chinese Romanized name into Chinese idiographic characters via corpus and web validation. In *Proceedings of CORIA 2005*, pp. 323-338, 2005.

M. Nagata, T. Saito, and K. Suzuki. Using the Web as a bilingual dictionary. 2001. In *Proceedings of 39th.* ACL Workshop on Data-Driven Methods in Machine *Translation*, pp. 95-102, 2001.

Y. Qu, and G. Grefenstette. 2004. Finding Ideographic Representations of Japanese Names Written in Latin Script via Language Identification and Corpus Validation. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, pp.183-190, 2004.

CK Quah. 2006. Translation and Technology, *Palgrave* Textbooks in Translation and Interpretation, Palgrave MacMillan.

R Sproat and C Shih. Statistical Method for Finding Word Boundaries in Chinese Text, Computer Processing of Chinese and Oriental languages. 1990.

J. C. Wu, T. Lin and J. S. Chang. Learning Source-Target Surface Patterns for Web-based Terminology Translation. In Proceeding of the ACL 2005 on Interactive poster and demonstration sessions (ACLdemo '05). 2005.

Y Zhang, F Huang, S Vogel. 2005. Mining translations of OOV terms from the web through cross-lingual query expansion. In Proceedings of the 28th Annual International ACM SIGIR, pp.669-670, 2005.

Beefmoves: Dissemination, Diversity, and Dynamics of English Borrowings in a German Hip Hop Forum

Matt Garley
Department of Linguistics
University of Illinois
707 S Mathews Avenue
Urbana, IL 61801, USA
mgarley2@illinois.edu

Julia Hockenmaier
Department of Computer Science
University of Illinois
201 N Goodwin Avenue
Urbana, IL 61801, USA
juliahmr@illinois.edu

Abstract

We investigate how novel English-derived words (anglicisms) are used in a German-language Internet hip hop forum, and what factors contribute to their uptake.

1 Introduction

Because English has established itself as something of a global lingua franca, many languages are currently undergoing a process of introducing new loanwords borrowed from English. However, while the motivations for borrowing are well studied, including e.g. the need to express concepts that do not have corresponding expressions in the recipient language, and the social prestige associated with the other language (Hock and Joseph, 1996), the dynamics of this process are poorly understood. While mainstream political debates often frame borrowing as evidence of cultural or linguistic decline, it is particularly pervasive in youth culture, which is often heavily influenced by North American trends. In many countries around the globe, hip hop fans form communities in which novel, creative uses of English are highly valued (Pennycook, 2007), indicative of group membership, and relatively frequent. We therefore study which factors contribute to the uptake of (hip hop-related) anglicisms in an online community of German hip hop fans over a span of 11 years.

2 The MZEE and Covo corpora

We collected a ∼12.5M word corpus (MZEE) of forum discussions from March 2000 to March 2011 on the German hip hop portal MZEE.com. A manual analysis of 10K words identified 8.2% of the tokens as anglicisms, contrasting with only 1.1% anglicisms in a major German news magazine, the *Spiegel* (Onysko, 2007, p.114). These anglicisms include uninflected English stems (e.g., *battle, rapper, flow*) as well as English stems with English inflection (e.g., *battled, rappers, flows*), English stems with German inflection (e.g., *gebattlet, rappern, flowen* 'battled, rappers, to flow'), and English stems with German derivational affixes (e.g., *battlemässig, rapperische, flowendere* 'battle-related, rapper-like, more flowing'), as well as compounds with one or more English parts (e.g., *battleraporientierter, hiphopgangstaghettorapper, maschinengewehrflow* 'someone oriented towards battle-rap, hip hop-gangsta-ghetto-rapper, machinegun flow'). We also collected a ∼20M word corpus (Covo) of English-language hip hop discussion (May 2003 - November 2011) from forums at ProjectCovo.com.

3 Identification of novel anglicisms

In order to identify novel anglicisms in the MZEE corpus, we have developed a classifier which can identify anglicism candidates, including those which incorporate German material (e.g., *möchtegerngangsterstyle* 'wannabe gangster style'), with very high recall. Since we are not interested in well-established anglicisms (e.g., *Baby*, *OK*), non-English words, or placenames, our goal is quite different from the standard language identification problem, including Alex (2008)'s inclusion classifier, which sought to identify 'foreign words' in general, including internationalisms, homographic

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 135–139,
Jeju, Republic of Korea, 8-14 July 2012.

Baseline n-gram classifier accuracy for n=

1	2	3	4	5	6	7
87.54	94.80	97.74	99.35	99.85	99.96	99.98

Figure 1: Accuracy of the baseline classifer on word lists; 10-fold CV; std. deviations ≤ 0.02 for all cases

words, and non-German placenames, but ignored hybrid/bilingual compounds and English words with German morphology during evaluation. Our final system consists of a binary classifier augmented with dictionary lookup for known words and two routines to deal with German morphology (affixation and compounding).

The baseline classifier We used MALLET (McCallum, 2002) to train a maximum entropy classifier, using character 1- through 6-grams (including word boundaries) as features. Since we could not manually annotate a large portion of the MZEE corpus, the training data consisted of the disjoint subsets of the English and German CELEX wordlists (Baayen et al., 1995), as well as the words used in Covo (to obtain coverage of hip hop English). We tested the classifier using 10-fold cross validation on the training data and on a manually annotated development set of 10K consecutive tokens from MZEE. All data was lowercased (this improved performance). We excluded from both data sets 4,156 words shared by the CELEX wordlists (such as Greek/Latin loanwoards common to both languages and homographs such as *hat*), 100 common German and 50 common English stop words, all 3-character words without vowels and 1,019 hip hop artists/label names, which reduced the development set from 10K tokens, or 3,380 distinct types, to 4,651 tokens and 2,741 types.

Affix-stripping Since German is a moderately inflected language, anglicisms are often 'hidden' by German morphology: in *geflowt* 'flowed', the English stem *flow* takes German participial affixes. We therefore included a template-based affix-stripping preprocessing step, removing common German affixes before feature extraction. Because of the possibility of multiple prefixation or suffixation (e.g. *rum-ge-battle* ('battling around') or *deep-er-en* ('deeper')), we stripped sequences of two prefixes and/or three suffixes. Our list of affixes was built

		Precision				
		All tokens		All types		OOVtyp.
Affix	Comp.	nodict	dict	nodict	dict	nodict
no	no	0.63	0.64	0.58	0.62	0.26
no	yes	**0.66**	0.69	0.58	0.62	0.27
yes	no	0.59	0.69	**0.60**	0.66	0.29
yes	yes	0.60	**0.70**	**0.60**	**0.67**	0.32

Table 1: Type- and token-based precision at recall=95

from commonly-affixed stems in the MZEE corpus and a German grammar (Fagan, 2009).

Compound-cutting Nominal and adjectival compounding is common in German, and loanword compounds are commonly found in MZEE:

(1) a. *chart|tauglich* ('suitable for the charts')
b. *flow|maschine|mässig* ('like a flow machine')
c. *Rap|vollpfosten* ('rap dumbasses')

Since these contain features that are highly indicative of German (e.g. *-lich#*, *ä*, and *pf*), we devised a compound-cutting procedure for words over length l (=7): if the word is initially classified as German, it is divided several ways according to the parameters n (=3), the number of cuts in each direction from the center, and m (=2), the minimum length of each part. Both halves are classified separately, and if the maximum anglicism classifier score out of all splits exceeds a target confidence c (=0.7), the original word is labeled a candidate anglicism. Parameter values were optimized on a subset of compounds from the development set.

Dictionary classification When applying the classifier to the MZEE corpus, words which occur exclusively in one of the German and English CELEX wordlists are automatically classified as such. This improved classifier results over tokens and types, as seen in Table 1 in the comparison of token and type precision for the dict/nodict conditions.

Evaluation We evaluated our system by adjusting the classifier threshold to obtain a recall level of 95% or higher on anglicism tokens in the development set (see Table 1). The final classifier achieved a per-token precision of 70% (per type: 67%) at 95% recall, a gain of 7% (9%) over the baseline.

Our system identified 1,415 anglicism candidate types with a corpus frequency of 100 or greater, out

of which we identified 851 (57.5%) for further investigation; 441 (31.1%) were either established anglicisms, place names, artist names, and other loanwords, and 123 (8.7%) were German words.

4 Predicting the fate of anglicisms

We examine here factors hypothesized to play a role in the establishment (or decline) of anglicisms.

Frequency in the English Covo corpus We first examine whether a word's frequency in the English-speaking hip hop community influences whether it becomes more frequently used in the German hip hop community. We aligned four large (>1M words each) 12-month time windows of the Covo and MZEE corpora, spanning the period 11-2003 through 11-2007. We used the 851 most frequent anglicisms identified in our system to find 106 English stems commonly used in German anglicisms, and compute their relative frequency (aggregated over all word forms) in each Covo and MZEE time window. We then measure correlation coefficients r between the frequency of a stem in Covo at time T_t, $f_t^E(stem)$, and the change in log frequency of the corresponding anglicisms in MZEE between T_t and a later time T_u, $\Delta \log_{10} f_{t:u}^G(w) = \log_{10} f_u^G(w) - \log_{10} f_t^G(w)$, as well as the corresponding p-values, and coefficients of determination R^2 (Table 2). There is a significant positive correlation between the variables, especially for change over a two-year time span.

Covo $\log_{10} f_t(stem)$ vs. MZEE $\Delta \log_{10} f_{t:u}(stem)$					
	r	p	t	R^2	N
$u = t + 1$ year	0.1891	*0.0007*	3.423	**3.6%**	318
$u = t + 2$ year	0.3130	*0.0001*	4.775	**9.8%**	212
$u = t + 3$ year	0.2327	*0.0164*	2.440	**5.4%**	106

Table 2: Correlations between stem frequency in Covo during year t and frequency change in MZEE between t and year $u = t + i$

Initial frequency and dissemination in MZEE In studying the fate of all words in two English Usenet corpora, Altmann, Pierrehumbert and Motter (2011, p.5) found that the measures D^U (dissemination over users) and D^T (dissemination over threads) predict changes in word frequency ($\Delta \log_{10} f$) better than initial word frequency ($\log_{10} f$). $D^U = \frac{U_w}{\tilde{U}_w}$ is defined as the ratio of the actual number of users of word w (U_w) over the expected number of users of w ($\tilde{U}_w$), and $D^T = \frac{T_w}{\tilde{T}_w}$ is calculated analogously fo the actual/expected number of threads in which w is used. $\tilde{U}_w$ and $\tilde{T}_w$ are estimated from a bag-of-words model approximating a Poisson process.

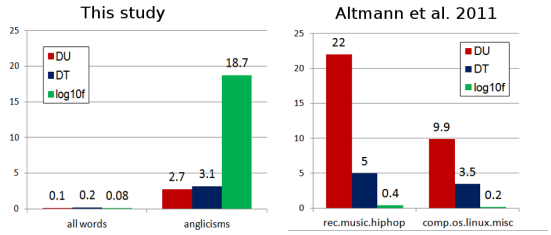


Figure 2: Correlation coefficient comparison of D_U, D_T, $\log_{10} f$ with $\Delta \log_{10} f$

We apply Altmann et al.'s model to study the difference in word dynamics between anglicisms and native words. Since we are not able to lemmatize the entire MZEE corpus, this study uses the 851 most common anglicism word forms identified by our system, treating all word forms as distinct. We split the MZEE corpus into six non-overlapping windows of 2M words each (T_1 through T_6), calculate $D_t^U(w)$, $D_t^T(w)$ and $\log_{10} f_t(w)$ within each time window T_t. We again measure how well these variables predict the change in log frequency $\Delta \log_{10} f_{t:u}(w) = \log_{10} f_u(w) - \log_{10} f_t(w)$ between the initial time T_t and a later time T_u, with $u = t + 1, ..., t + 3$.

When measured over all words excluding anglicisms, $\log_{10} f_t$, D_t^U, and D_t^T at an initial time are very weakly ($0.0309 < r < 0.0692$), but significantly ($p < .0001$) positively correlated with $\Delta \log_{10} f_{t:u}$. However, in contrast to Altmann et al.'s findings that D^U and D^T serve better than frequency as predictors of word fate, for the set of anglicisms (Table 3), all correlations were both negative and stronger, and initial frequency $\log_{10} f_t$ (not dissemination) is the best predictor, especially as the time spans increase in length. That is, while most words' frequency change cannot generally be predicted from earlier frequency, we find that, for anglicisms, a high frequency is more likely to lead to a decline, and vice versa.[1].

[1]A set of 337 native German words frequency-matched to the most common 337 anglicisms in our data set patterns with the superset of all words (i.e., is not well predicted by any of the

$\Delta \log_{10} f_{t:t+1}(w)$					
	r	p	t	R^2	N
$\log_{10} f_t$	-0.2919	*<.0001*	-19.641	**8.5%**	4145
D_t^U	-0.0814	*.0001*	-5.258	**0.7%**	4145
D_t^T	-0.0877	*.0001*	-5.668	**0.8%**	4145
$\Delta \log_{10} f_{t:t+2}(w)$					
$\log_{10} f_t$	-0.3580	*<.0001*	-22.042	**12.8%**	3306
D_t^U	-0.1207	*.0001*	-6.987	**1.5%**	3306
D_t^T	-0.1373	*.0001*	-7.97	**1.9%**	3306
$\Delta \log_{10} f_{t:t+3}(w)$					
$\log_{10} f_t$	-0.4329	*<.0001*	-23.864	**18.7%**	2471
D_t^U	-0.1634	*.0001*	-8.229	**2.7%**	2471
D_t^T	-0.1755	*.0001*	-8.858	**3.1%**	2471

Table 3: Correlations between initial frequency and dissemination over users and threads and a change in frequency for the 851 most common anglicisms in MZEE.

Finally, from the comparison of timespans in Table 3, we see that the predictive ability (R^2) of the three measures increases as the timespan for $\Delta \log_{10} f$ becomes longer, i.e., frequency and dissemination effects on frequency change do not operate as strongly in immediate time scales.[2].

5 Conclusion

In this study, we examined factors hypothesized to influence the propagation of words through a community of speakers, focusing on anglicisms in a German hip hop discussion corpus. The first analysis presented here sheds light on the lexical dynamics between the English and German hip hop communities, demonstrating that English frequency correlates positively with change in a borrowed word's frequency in the German community–this result is not shocking, as the communities are exposed to shared inputs (e.g., hip hop lyrics), but the strength of this correlation is highest in a two-year timespan, suggesting a time lag from the frequency of hip hop terms in English to the effects on those terms in German. Future research here could profitably focus on this relationship, especially for terms whose success in the English and German hip hop communities is highly disparate. Investigation of those terms could suggest non-frequency factors which affect a word's success or failure.

The second analysis, which compared three measures used by Altmann, Pierrehumbert, and Motter (2011) to predict lexical frequency change, found that $\log_{10} f$, D^U, and D^T did not predict frequency change well for non-anglicism words in the MZEE corpus, but that $\log_{10} f$ in particular does predict frequency change for anglicisms, though this correlation is inverse; this finding relates to another analysis of loanwords. In a diachronic study of loanword frequencies in two French newspaper corpora, Chesley and Baayen (2010, p.1364-5) found that high initial frequency was "a bad omen for a borrowing" and found an interaction effect between frequency and dispersion (roughly equivalent to dissemination in the present study): "As dispersion and frequency increase, the number of occurrences at T2 decreases."

A view of language as a stylistic resource (Coupland, 2007) provides some explanation for these counter-intuitive findings: An anglicism which is used less often initially but survives is likely to increase in frequency as other speakers adopt it for 'cred' or in-group prestige. However, a highly frequent anglicism seems to become increasingly undesirable–after all, if everyone is using it, it loses its capacity to distinguish in-group members (consider, e.g., the widespread adoption of the term *bling* outside hip hop culture in the US). This circumstance is reflected by a drop in frequency as the word becomes passé. This view is supported by ethnographic interviews with members of the German hip hop community: *"Yeah, [the use of anglicisms is] naturally overdone, for the most part. It's targeted at these 15, 14-year-old kids, that think this is cool. The* crowd*! Ah, cool! Yeah, it's true–the* crowd*, even I say that, but not seriously."* -'Peter', 22, beatboxer and student at the Hip Hop Academy Hamburg.

In summary, the analyses discussed here leverage the opportunities provided by large-scale corpus analysis and by the uniquely language-focused nature of the hip hop community to investigate issues of sociohistorical linguistic concern: what sort of factors are at work in the process of linguistic change through contact, and more specifically, which word-extrinsic properties of stems and wordforms condition the success and failure of borrowed English words in the German hip hop community.

variables) in this regard.

[2]An analysis which truncated the forms in the first two timespans to match the N of the third confirm that this increase is not simply an effect of the number of cases considered.

Acknowledgements

Matt Garley was supported by the Cognitive Science/Artificial Intelligence Fellowship from the University of Illinois and a German Academic Exchange Service (DAAD) Graduate Research Grant. Julia Hockenmaier is supported by the National Science Foundation through CAREER award 1053856 and award 0803603. The authors would like to thank Dr. Marina Terkourafi of the University of Illinois at Urbana-Champaign Linguistics Department for her insights and contributions to this research project.

References

Beatrice Alex. 2008. *Automatic detection of English inclusions in mixed-lingual data with an application to parsing*. Ph.D. thesis, Institute for Communicating and Collaborative Systems, School of Informatics, University of Edinburgh.

Eduardo G. Altmann, Janet B. Pierrehumbert, and Adilson E. Motter. 2011. Niche as a determinant of word fate in online groups. *PLoS ONE*, 6(5):e19009, 05.

R.H. Baayen, R. Piepenbrock, and L. Gulikers. 1995. The CELEX lexical database. CD-ROM.

Paula Chesley and R.H. Baayen. 2010. Predicting new words from newer words: Lexical borrowings in french. *Linguistics*, 45(4):1343–1374.

Nikolas Coupland. 2007. *Style: Language variation and identity*. Cambridge, UK: Cambridge University Press.

Sarah M.B. Fagan. 2009. *German: A linguistic introduction*. Cambridge, UK: Cambridge University Press.

Hans Henrich Hock and Brian D. Joseph. 1996. *Language history, language change, and language relationship: An introduction to historical and comparative linguistics*. Berlin, New York: Mouton de Gruyter.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. Web: http://mallet.cs.umass.edu.

Alexander Onysko. 2007. *Anglicisms in German: Borrowing, lexical productivity, and written codeswitching*. Berlin: Walter de Gruyter.

Alastair Pennycook. 2007. *Global Englishes and transcultural flows*. New York, London: Routledge.

Learning the Latent Semantics of a Concept from its Definition

Weiwei Guo
Department of Computer Science,
Columbia University,
New York, NY, USA
weiwei@cs.columbia.edu

Mona Diab
Center for Computational Learning Systems,
Columbia University,
New York, NY, USA
mdiab@ccls.columbia.edu

Abstract

In this paper we study unsupervised word sense disambiguation (WSD) based on sense definition. We learn low-dimensional latent semantic vectors of concept definitions to construct a more robust sense similarity measure *wmfvec*. Experiments on four all-words WSD data sets show significant improvement over the baseline WSD systems and LDA based similarity measures, achieving results comparable to state of the art WSD systems.

1 Introduction

To date, many unsupervised WSD systems rely on a sense similarity module that returns a similarity score given two senses. Many similarity measures use the taxonomy structure of WordNet [WN] (Fellbaum, 1998), which allows only noun-noun and verb-verb pair similarity computation since the other parts of speech (adjectives and adverbs) do not have a taxonomic representation structure. For example, the *jcn* similarity measure (Jiang and Conrath, 1997) computes the sense pair similarity score based on the information content of three senses: the two senses and their least common subsumer in the noun/verb hierarchy.

The most popular sense similarity measure is the Extended Lesk [*elesk*] measure (Banerjee and Pedersen, 2003). In *elesk*, the similarity score is computed based on the length of overlapping words/phrases between two extended dictionary definitions. The definitions are extended by definitions of neighbor senses to discover more overlapping words. However, exact word matching is lossy. Below are two definitions from WN:

bank#n#1: *a financial institution that accepts deposits and channels the money into lending activities*

stock#n#1: *the capital raised by a corporation through the issue of shares entitling holders to an ownership interest (equity)*

Despite the high semantic relatedness of the two senses, the overlapping words in the two definitions are only *a, the*, leading to a very low similarity score.

Accordingly we are interested in extracting latent semantics from sense definitions to improve *elesk*. However, the challenge lies in that sense definitions are typically too short/sparse for latent variable models to learn accurate semantics, since these models are designed for long documents. For example, topic models such as LDA (Blei et al., 2003), can only find the dominant topic based on the observed words in a definition ($financial$ topic in $bank\#n\#1$ and $stock\#n\#1$) without further discernibility. In this case, many senses will share the same latent semantics profile, as long as they are in the same topic/domain.

To solve the sparsity issue we use missing words as negative evidence of latent semantics, as in (Guo and Diab, 2012). We define missing words of a sense definition as the whole vocabulary in a corpus minus the observed words in the sense definition. Since observed words in definitions are too few to reveal the semantics of senses, missing words can be used to tell the model what the definition is **not** about. Therefore, we want to find a latent semantics profile that is related to observed words in a definition, but also **not** related to missing words, so that the induced latent semantics is unique for the sense.

Finally we also show how to use WN neighbor sense definitions to construct a nuanced sense similarity *wmfvec*, based on the inferred latent semantic vectors of senses. We show that *wmfvec* outperforms *elesk* and LDA based approaches in four All-words WSD data sets. To our best knowledge, *wmfvec* is the first sense similarity measure based on latent semantics of sense definitions.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 140–144,
Jeju, Republic of Korea, 8-14 July 2012.

	financial	sport	institution	R_o	R_m
v_1	1	0	0	20	600
v_2	0.6	0	0.1	18	300
v_3	0.2	0.3	0.2	5	100

Table 1: Three possible hypotheses of latent vectors for the definition of $bank\#n\#1$

2 Learning Latent Semantics of Definitions

2.1 Intuition

Given only a few observed words in a definition, there are many hypotheses of latent vectors that are highly related to the observed words. Therefore, missing words can be used to prune the hypotheses that are also highly related to the missing words.

Consider the hypotheses of latent vectors in table 1 for $bank\#n\#1$. Assume there are 3 dimensions in our latent model: *financial, sport, 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. Hypothesis v_1 is given by topic models, where only the $financial$ dimension is found, and it has the maximum relatedness to observed words in $bank\#n\#1$ definition $R_o^{v_1} = 20$. v_2 is the ideal latent vector, since it also detects that $bank\#n\#1$ is related to $institution$. It has a slightly smaller $R_o^{v_2} = 18$, but more importantly, its relatedness to missing words, $R_m^{v_2} = 300$, is substantially smaller than $R_m^{v_1} = 600$.

However, we cannot simply choose a hypothesis with the maximum $R_o - R_m$ value, since v_3, which is clearly not related to $bank\#n\#1$ but with a minimum $R_m = 100$, will therefore be (erroneously) returned as the answer. The solution is straightforward: give a smaller weight to missing words, e.g., so that the algorithm tries to select a hypothesis with maximum value of $R_o - 0.01 \times R_m$. We choose weighted matrix factorization [WMF] (Srebro and Jaakkola, 2003) to implement this idea.

2.2 Modeling Missing Words by Weighted Matrix Factorization

We represent the corpus of WN definitions as an $M \times N$ matrix X, where row entries are M unique words existing in WN definitions, and columns represent N WN sense ids. The cell X_{ij} records the TF-IDF value of word w_i appearing in definition of sense s_j.

In WMF, the original matrix X is factorized into two matrices such that $X \approx P^\top Q$, where P is a $K \times M$ matrix, and Q is a $K \times N$ matrix. In this scenario, the latent semantics of each word w_i or sense s_j is represented as a K-dimension vector $P_{\cdot,i}$ or $Q_{\cdot,j}$ respectively. Note that the inner product of $P_{\cdot,i}$ and $Q_{\cdot,j}$ is used to approximate the semantic relatedness of word w_i and definition of sense s_j: $X_{ij} \approx P_{\cdot,i} \cdot Q_{\cdot,j}$.

In WMF each cell is associated with a weight, so missing words cells (X_{ij}=0) can have a much less contribution than observed words. Assume w_m is the weight for missing words cells. The latent vectors of words P and senses Q are estimated by minimizing the objective function:[1]

$$\sum_i \sum_j W_{ij} \left(P_{\cdot,i} \cdot Q_{\cdot,j} - X_{ij}\right)^2 + \lambda ||P||_2^2 + \lambda ||Q||_2^2$$
$$\text{where } W_{i,j} = \begin{cases} 1, & \text{if } X_{ij} \neq 0 \\ w_m, & \text{if } X_{ij} = 0 \end{cases} \quad (1)$$

Equation 1 explicitly requires the latent vector of sense $Q_{\cdot,j}$ to be not related to missing words ($P_{\cdot,i} \cdot Q_{\cdot,j}$ should be close to 0 for missing words $X_{ij} = 0$). Also weight w_m for missing words is very small to make sure latent vectors such as v_3 in table 1 will not be chosen. In experiments we set $w_m = 0.01$.

After we run WMF on the definitions corpus, the similarity of two senses s_j and s_k can be computed by the inner product of $Q_{\cdot,j}$ and $Q_{\cdot,k}$.

2.3 A Nuanced Sense Similarity: *wmfvec*

We can further use the features in WordNet to construct a better sense similarity measure. The most important feature of WN is senses are connected by relations such as *hypernymy, meronymy, similar attributes*, etc. We observe that neighbor senses are usually similar, hence they could be a good indicator for the latent semantics of the target sense.

We use WN neighbors in a way similar to *elesk*. Note that in *elesk* each definition is extended by including definitions of its neighbor senses. Also, they do not normalize the length. In our case, we also adopt these two ideas: (1) a sense is represented by the sum of its original latent vector and its neighbors' latent vectors. Let $N(j)$ be the set of neighbor senses of sense j. then new latent vector is: $Q_{\cdot,j}^{new} = Q_{\cdot,j} + \sum_k^{k \in N(j)} Q_{\cdot,k}$ (2) Inner product (instead of cosine similarity) of the two resulting sense vectors is treated as the sense pair similarity. We refer to our sense similarity measure as *wmfvec*.

[1] Due to limited space inference and update rules for P and Q are omitted, but can be found in (Srebro and Jaakkola, 2003)

3 Experiment Setting

Task: We choose the fine-grained All-Words Sense Disambiguation task, where systems are required to disambiguate all the content words (noun, adjective, adverb and verb) in documents. The data sets we use are all-words tasks in SENSEVAL2 [SE2], SENSEVAL3 [SE3], SEMEVAL-2007 [SE07], and Semcor. We tune the parameters in *wmfvec* and other baselines based on SE2, and then directly apply the tuned models on other three data sets.

Data: The sense inventory is WN3.0 for the four WSD data sets. WMF and LDA are built on the corpus of sense definitions of two dictionaries: WN and Wiktionary [Wik].[2] We do not link the senses across dictionaries, hence Wik is only used as augmented data for WMF to better learn the semantics of words. All data is tokenized, POS tagged (Toutanova et al., 2003) and lemmatized, resulting in 341,557 sense definitions and 3,563,649 words.

WSD Algorithm: To perform WSD we need two components: (1) a sense similarity measure that returns a similarity score given two senses; (2) a disambiguation algorithm that determines which senses to choose as final answers based on the sense pair similarity scores. We choose the Indegree algorithm used in (Sinha and Mihalcea, 2007; Guo and Diab, 2010) as our disambiguation algorithm. It is a graph-based algorithm, where nodes are senses, and edge weight equals to the sense pair similarity. The final answer is chosen as the sense with maximum indegree. Using the Indegree algorithm allows us to easily replace the sense similarity with *wmfvec*. In Indegree, two senses are connected if their words are within a local window. We use the optimal window size of 6 tested in (Sinha and Mihalcea, 2007; Guo and Diab, 2010).

Baselines: We compare with (1) *elesk*, the most widely used sense similarity. We use the implementation in (Pedersen et al., 2004).

We believe WMF is a better approach to model latent semantics than LDA, hence the second baseline (2) LDA using Gibbs sampling (Griffiths and Steyvers, 2004). However, we cannot directly use estimated topic distribution $P(z|d)$ to represent the definition since it only has non-zero values on one or two topics. Instead, we calculate the latent vector of a definition by summing up the $P(z|w)$ of all constituent words weighted by X_{ij}, which gives much better WSD results.[3] We produce LDA vectors [*ldavec*] in the same setting as *wmfvec*, which means it is trained on the same corpus, uses WN neighbors, and is tuned on SE2.

[2] http://en.wiktionary.org/

Data	Model	Total	Noun	Adj	Adv	Verb
SE2	random	40.7	43.9	43.6	58.2	21.6
	elesk	56.0	63.5	63.9	62.1	30.8
	ldavec	58.6	68.6	60.2	66.1	33.2
	wmfvec	**60.5**	**69.7**	**64.5**	**67.1**	**34.9**
	jcn+elesk	60.1	69.3	63.9	62.8	37.1
	jcn+wmfvec	**62.1**	**70.8**	**64.5**	**67.1**	**39.9**
SE3	random	33.5	39.9	44.1	-	33.5
	elesk	52.3	58.5	57.7	-	41.4
	ldavec	53.5	58.1	60.8	-	43.7
	wmfvec	**55.8**	**61.5**	**64.4**	-	**43.9**
	jcn+elesk	55.4	60.5	57.7	-	47.4
	jcn+wmfvec	**57.4**	**61.2**	**64.4**	-	**48.8**
SE07	random	25.6	27.4	-	-	24.6
	elesk	42.2	47.2	-	-	39.5
	ldavec	43.7	49.7	-	-	40.5
	wmfvec	**45.1**	**52.2**	-	-	**41.2**
	jcn+elesk	44.5	52.8	-	-	40.0
	jcn+wmfvec	**45.5**	**53.5**	-	-	**41.2**
Semcor	random	35.26	40.13	50.02	58.90	20.08
	elesk	55.43	61.04	69.30	62.85	43.36
	ldavec	58.17	63.15	70.08	**67.97**	46.91
	wmfvec	**59.10**	**64.64**	**71.44**	67.05	**47.52**
	jcn+elesk	61.61	69.61	69.30	62.85	50.72
	jcn+wmfvec	**63.05**	**70.64**	**71.45**	**67.05**	**51.72**

Table 2: WSD results per POS ($K = 100$)

At last, we compare *wmfvec* with a mature WSD system based on sense similarities, (3) (Sinha and Mihalcea, 2007) [*jcn+elesk*], where they evaluate six sense similarities, select the best of them and combine them into one system. Specifically, in their implementation they use *jcn* for noun-noun and verb-verb pairs, and *elesk* for other pairs. (Sinha and Mihalcea, 2007) used to be the state-of-the-art system on SE2 and SE3.

4 Experiment Results

The disambiguation results ($K = 100$) are summarized in Table 2. We also present in Table 3 results using other values of dimensions K for *wmfvec* and *ldavec*. There are very few words that are not covered due to failure of lemmatization or POS tag mismatches, thereby F-measure is reported.

Based on SE2, *wmfvec*'s parameters are tuned as $\lambda = 20, w_m = 0.01$; *ldavec*'s parameters are tuned as $\alpha = 0.05, \beta = 0.05$. We run WMF on WN+Wik for 30 iterations, and LDA for 2000 iterations. For

[3] It should be noted that this renders LDA a very challenging baseline to outperform.

LDA, more robust $P(w|z)$ is generated by averaging over the last 10 sampling iterations. We also set a threshold to *elesk* similarity values, which yields better performance. Same as (Sinha and Mihalcea, 2007), values of *elesk* larger than 240 are set to 1, and the rest are mapped to [0,1].

elesk* vs *wmfvec: *wmfvec* outperforms *elesk* consistently in all POS cases (noun, adjective, adverb and verb) on four datasets by a large margin ($2.9\% - 4.5\%$ in $total$ case). Observing the results yielded per POS, we find a large improvement comes from nouns. Same trend has been reported in other distributional methods based on word co-occurrence (Cai et al., 2007; Li et al., 2010; Guo and Diab, 2011). More interestingly, *wmfvec* also improves verbs accuracy significantly.

ldavec* vs *wmfvec: *ldavec* also performs very well, again proving the superiority of latent semantics over surface words matching. However, *wmfvec* also outperforms *ldavec* in every POS case except Semcor adverbs (at least $+1\%$ in $total$ case). We observe the trend is consistent in Table 3 where different dimensions are used for *ldavec* and *wmfvec*. These results show that given the same text data, WMF outperforms LDA on modeling latent semantics of senses by exploiting missing words.

***jcn+elesk* vs *jcn+wmfvec*:** *jcn+elesk* is a very mature WSD system that takes advantage of the great performance of *jcn* on noun-noun and verb-verb pairs. Although *wmfvec* does much better than *elesk*, using *wmfvec* solely is sometimes outperformed by *jcn+elesk* on nouns and verbs. Therefore to beat *jcn+elesk*, we replace the *elesk* in *jcn+elesk* with *wmfvec* (hence *jcn+wmfvec*). Similar to (Sinha and Mihalcea, 2007), we normalize *wmfvec* similarity such that values greater than 400 are set to 1, and the rest values are mapped to [0,1]. We choose the value 400 based on the WSD performance on tuning set SE2. As expected, the resulting *jcn+wmfvec* can further improve *jcn+elesk* for all cases. Moreover, *jcn+wmfvec* produces similar results to state-of-the-art unsupervised systems on SE02, 61.92% F-mearure in (Guo and Diab, 2010) using WN1.7.1, and SE03, 57.4% in (Agirre and Soroa, 2009) using WN1.7. It shows *wmfvec* is robust that it not only performs very well individually, but also can be easily incorporated with existing evidence as represented using *jcn*.

dim	SE2	SE3	SE07	Semcor
50	57.4 - **60.5**	52.9 - 54.9	43.1 - 44.2	57.90 - 58.99
75	57.8 - 60.3	53.5 - 55.2	43.3 - 44.6	58.12 - 59.07
100	58.6 - **60.5**	53.5 - **55.8**	43.7 - 45.1	58.17 - 59.10
125	58.2 - 60.2	53.9 - 55.5	43.7 - 45.1	58.26 - **59.19**
150	58.2 - 59.8	53.6 - 54.6	44.4 - **45.9**	58.13 - 59.15

Table 3: *ldavec* and *wmfvec* (latter) results per # of dimensions

4.1 Discussion

We look closely into WSD results to obtain an intuitive feel for what is captured by *wmfvec*. For example, the target word *mouse* in the context: *... in experiments with mice that a gene called p53 could transform normal cells into cancerous ones... elesk* returns the wrong sense *computer device*, due to the sparsity of overlapping words between definitions of *animal mouse* and the context words. *wmfvec* chooses the correct sense *animal mouse*, by recognizing the biology element of *animal mouse* and related context words *gene, cell, cancerous*.

5 Related Work

Sense similarity measures have been the core component in many unsupervised WSD systems and lexical semantics research/applications. To date, *elesk* is the most popular such measure (McCarthy et al., 2004; Mihalcea, 2005; Brody et al., 2006). Sometimes people use *jcn* to obtain similarity of noun-noun and verb-verb pairs (Sinha and Mihalcea, 2007; Guo and Diab, 2010). Our similarity measure *wmfvec* exploits the same information (sense definitions) *elesk* and *ldavec* use, and outperforms them significantly on four standardized data sets. To our best knowledge, we are the first to construct a sense similarity by latent semantics of sense definitions.

6 Conclusions

We construct a sense similarity *wmfvec* from the latent semantics of sense definitions. Experiment results show *wmfvec* significantly outperforms previous definition-based similarity measures and LDA vectors on four all-words WSD data sets.

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

Eneko Agirre and Aitor Soroa. 2009. Proceedings of personalizing pagerank for word sense disambiguation. In *the 12th Conference of the European Chapter of the ACL*.

Satanjeev Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 805–810.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3.

Samuel Brody, Roberto Navigli, and Mirella Lapata. 2006. Ensemble methods for unsupervised wsd. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*.

Jun Fu Cai, Wee Sun Lee, and Yee Whye Teh. 2007. Improving word sense disambiguation using topic features. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101.

Weiwei Guo and Mona Diab. 2010. Combining orthogonal monolingual and multilingual sources of evidence for all words wsd. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

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*.

Weiwei Guo and Mona Diab. 2012. Modeling sentences in the latent space. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*.

Jay J. Jiang and David W. Conrath. 1997. Finding predominant word senses in untagged text. In *Proceedings of International Conference Research on Computational Linguistics*.

Linlin Li, Benjamin Roth, and Caroline Sporleder. 2010. Topic models for word sense disambiguation and token-based idiom detection. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding predominant word senses in untagged text. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics*.

Rada Mihalcea. 2005. Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of the Joint Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 411–418.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity - measuring the relatedness of concepts. In *Proceedings of Fifth Annual Meeting of the North American Chapter of the Association for Computational Linguistics*.

Ravi Sinha and Rada Mihalcea. 2007. Unsupervised graph-based word sense disambiguation using measures of word semantic similarity. In *Proceedings of the IEEE International Conference on Semantic Computing*, pages 363–369.

Nathan Srebro and Tommi Jaakkola. 2003. Weighted low-rank approximations. In *Proceedings of the Twentieth International Conference on Machine Learning*.

Kristina Toutanova, Dan Klein, Christopher 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*.

Unsupervised Semantic Role Induction with Global Role Ordering

Nikhil Garg
University of Geneva
Switzerland
nikhil.garg@unige.ch

James Henderson
University of Geneva
Switzerland
james.henderson@unige.ch

Abstract

We propose a probabilistic generative model for unsupervised semantic role induction, which integrates local role assignment decisions and a global role ordering decision in a unified model. The role sequence is divided into *intervals* based on the notion of *primary roles*, and each interval generates a sequence of *secondary roles* and syntactic constituents using local features. The global role ordering consists of the sequence of primary roles only, thus making it a partial ordering.

1 Introduction

Unsupervised semantic role induction has gained significant interest recently (Lang and Lapata, 2011b) due to limited amounts of annotated corpora. A Semantic Role Labeling (SRL) system should provide consistent argument labels across different syntactic realizations of the same verb (Palmer et al., 2005), as in

$(a.)$ [Mark]$_{A0}$ drove [the car]$_{A1}$

$(b.)$ [The car]$_{A1}$ was driven by [Mark]$_{A0}$

This simple example also shows that while certain local syntactic and semantic features could provide clues to the semantic role label of a constituent, non-local features such as predicate voice could provide information about the expected semantic role sequence. Sentence a is in active voice with sequence ($A0$, $PREDICATE$, $A1$) and sentence b is in passive voice with sequence ($A1$, $PREDICATE$, $A0$). Additional global preferences, such as arguments $A0$ and $A1$ rarely repeat in a frame (as seen in the corpus), could also be useful in addition to local features.

Supervised SRL systems have mostly used local classifiers that assign a role to each constituent independently of others, and only modeled limited correlations among roles in a sequence (Toutanova et al., 2008). The correlations have been modeled via *role sets* (Gildea and Jurafsky, 2002), role repetition constraints (Punyakanok et al., 2004), language model over roles (Thompson et al., 2003; Pradhan et al., 2005), and global role sequence (Toutanova et al., 2008). Unsupervised SRL systems have explored even fewer correlations. Lang and Lapata (2011a; 2011b) use the relative position (left/right) of the argument w.r.t. the predicate. Grenager and Manning (2006) use an ordering of the linking of semantic roles and syntactic relations. However, as the space of possible linkings is large, language-specific knowledge is used to constrain this space.

Similar to Toutanova et al. (2008), we propose to use global role ordering preferences but in a generative model in contrast to their discriminative one. Further, unlike Grenager and Manning (2006), we do not explicitly generate the linking of semantic roles and syntactic relations, thus keeping the parameter space tractable. The main contribution of this work is an unsupervised model that uses global role ordering and repetition preferences without assuming any language-specific constraints.

Following Gildea and Jurafsky (2002), previous work has typically broken the SRL task into (i) argument identification, and (ii) argument classification (Màrquez et al., 2008). The latter is our focus in this work. Given the dependency parse tree of a sentence with correctly identified arguments, the aim is to assign a semantic role label to each argument.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 145–149,
Jeju, Republic of Korea, 8-14 July 2012.

Algorithm 1 Generative process

——————— PARAMETERS ———————
for all predicate p **do**
  for all voice $vc \in \{active, passive\}$ **do**
    draw $\theta^{order}_{p,vc} \sim Dirichlet(\alpha^{order})$
  for all interval I **do**
    draw $\theta^{SR}_{p,I} \sim Dirichlet(\alpha^{SR})$
    for all adjacency $adj \in \{0, 1\}$ **do**
      draw $\theta^{STOP}_{p,I,adj} \sim Beta(\alpha^{STOP})$
  for all role $r \in PR \cup SR$ **do**
    for all feature type f **do**
      draw $\theta^{F}_{p,r,f} \sim Dirichlet(\alpha^{F})$
——————— DATA ———————
given a predicate p with voice vc:
choose an ordering $o \sim Multinomial(\theta^{order}_{p,vc})$
for all interval $I \in o$ **do**
  draw an indicator $s \sim Binomial(\theta^{STOP}_{p,I,0})$
  while $s \neq STOP$ **do**
    choose a SR $r \sim Multinomial(\theta^{SR}_{p,I})$
    draw an indicator $s \sim Binomial(\theta^{STOP}_{p,I,1})$
for all generated roles r **do**
  for all feature type f **do**
    choose a value $v_f \sim Multinomial(\theta^{F}_{p,r,f})$

2 Proposed Model

We assume the roles to be predicate-specific. We begin by introducing a few terms:

Primary Role (PR) For every predicate, we assume the existence of K primary roles (PRs) denoted by $P_1, P_2, ..., P_K$. These roles are not allowed to repeat in a frame and serve as "anchor points" in the global role ordering. Intuitively, the model attempts to choose PRs such that they occur with high frequency, do not repeat, and their ordering influences the positioning of other roles. Note that a PR may correspond to either a core role or a modifier role. For ease of explication, we create 3 additional PRs: $START$ denoting the start of the role sequence, END denoting its end, and $PRED$ denoting the predicate.

Secondary Role (SR) The roles that are not PRs are called secondary roles (SRs). Given N roles in total, there are $(N - K)$ SRs, denoted by $S_1, S_2, ..., S_{N-K}$. Unlike PRs, SRs are not constrained to occur only once in a frame and do not participate in the global role ordering.

Interval An interval is a sequence of SRs bounded by PRs, for instance $(P_2, S_3, S_5, PRED)$.

Ordering An ordering is the sequence of PRs observed in a frame. For example, if the complete role sequence is $(START, P_2, S_1, S_1, PRED, S_3, END)$, the ordering is defined as $(START, P_2, PRED, END)$.

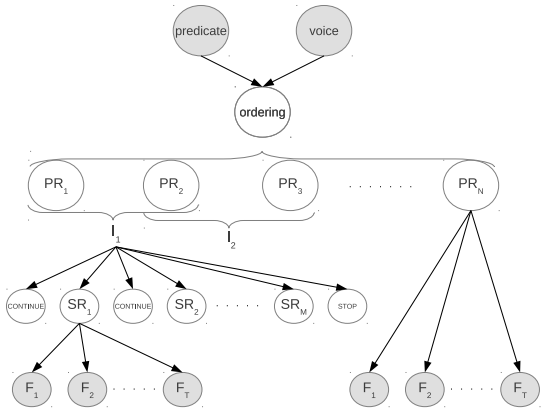


Figure 1: Proposed model. Shaded and unshaded nodes represent visible and hidden variables resp.

Features We have explored 1 frame level (global) feature (i) *voice*: active/passive, and 3 argument level (local) features (i) *deprel*: dependency relation of an argument to its head in the dependency parse tree, (ii) *head*: head word of the argument, and (iii) *pos-head*: Part-of-Speech tag of *head*.

Algorithm 1 describes the generative story of our model and Figure 1 illustrates it graphically. Given a predicate and its voice, an ordering is selected from a multinomial. This ordering gives us the sequence of PRs $(PR_1, PR_2, ..., PR_N)$. Each pair of consecutive PRs, PR_i, PR_{i+1}, in an ordering corresponds to an interval I_i. For each such interval, we generate 0 or more SRs $(SR_{i1}, SR_{i2}, ...SR_{iM})$ as follows. Generate an indicator variable: $CONTINUE/STOP$ from a binomial distribution. If $CONTINUE$, generate a SR from the multinomial corresponding to the interval. Generate another indicator variable and continue the process till a $STOP$ has been generated. In addition to the interval, the indicator variable also depends on whether we are generating the first SR ($adj = 0$) or a subsequent one ($adj = 1$). For each role, primary as well as secondary, we now generate the corresponding constituent by generating each of its features independently $(F_1, F_2, ..., F_T)$.

Given a frame instance with predicate p and voice vc, Figure 2 gives (i) Eq. 1: the joint distribution of the ordering o, role sequence $\mathbf{r}$, and constituent sequence $\mathbf{f}$, and (ii) Eq. 2: the marginal distribution of an instance. The likelihood of the whole corpus is the product of marginals of individual instances.

$$P(o, \mathbf{r}, \mathbf{f}|p, vc) = \underbrace{P(o|p, vc)}_{\text{ordering}} * \underbrace{\Pi_{\{r_i \in \mathbf{r} \cap PR\}} P(f_i|r_i, p)}_{\text{Primary Roles}} * \underbrace{\Pi_{\{I \in o\}} P(\mathbf{r}(I), \mathbf{f}(I)|I, p)}_{\text{Intervals}} \quad (1)$$

$$\text{where} \quad P(\mathbf{r}(I), \mathbf{f}(I)|I, p) = \prod_{r_i \in \mathbf{r}(I)} \underbrace{P(continue|I, p, adj)}_{\text{generate indicator}} \underbrace{P(r_i|I, p)}_{\text{generate SR}} \underbrace{P(f_i|r_i, p)}_{\text{generate features}} * \underbrace{P(stop|I, p, adj)}_{\text{end of the interval}}$$

$$\text{and} \quad P(f_i|r_i, p) = \Pi_t P(f_{i,t}|r_i, p)$$

$$P(\mathbf{f}|p, vc) = \Sigma_o \Sigma_{\{\mathbf{r} \in seq(o)\}} P(o, \mathbf{r}, \mathbf{f}|p, vc) \qquad \text{where } seq(o) = \{\text{role sequences allowed under ordering } o\} \quad (2)$$

Figure 2: r_i and f_i denote the role and features at position i respectively, and $\mathbf{r}(I)$ and $\mathbf{f}(I)$ respectively denote the SR sequence and feature sequence in interval I. $f_{i,t}$ denotes the value of feature t at position i.

This particular choice of model is inspired from different sources. Firstly, making the role ordering dependent only on PRs aligns with the observation by Pradhan et al. (2005) and Toutanova et al. (2008) that including the ordering information of only core roles helped improve the SRL performance as opposed to the complete role sequence. Although our assumption here is softer in that we assume the existence of some roles which define the ordering which may or may not correspond to core roles. Secondly, generating the SRs independently of each other given the interval is based on the intuition that knowing the core roles informs us about the expected non-core roles that occur between them. This intuition is supported by the statistics in the annotated data, where we found that if we consider the core roles as PRs, then most of the intervals tend to have only a few types of SRs and a given SR tends to occur only in a few types of intervals. The concept of intervals is also related to the linguistic theory of topological fields (Diderichsen, 1966; Drach, 1937). This simplifying assumption that given the PRs at the interval boundary, the SRs in that interval are independent of the other roles in the sequence, keeps the parameter space limited, which helps unsupervised learning. Thirdly, not allowing some or all roles to repeat has been employed as a useful constraint in previous work (Punyakanok et al., 2004; Lang and Lapata, 2011b), which we use here for PRs. Lastly, conditioning the ($STOP/CONTINUE$) indicator variable on the adjacency value (adj) is inspired from the DMV model (Klein and Manning, 2004) for unsupervised dependency parsing. We found in the annotated corpus that if we map core roles to PRs, then most of the time the intervals do not generate any SRs at all. So, the probability to $STOP$ should be very high when generating the first SR.

We use an EM procedure to train the model. In the E-step, we calculate the expected counts of all the hidden variables in our model using the Inside-Outside algorithm (Baker, 1979). In the M-step, we add the counts corresponding to the Bayesian priors to the expected counts and use the resulting counts to calculate the MAP estimate of the parameters.

3 Experiments

Following the experimental settings of Lang and Lapata (2011b), we use the CoNLL 2008 shared task dataset (Surdeanu et al., 2008), only consider verbal predicates, and run unsupervised training on the standard training set. The evaluation measures are also the same: (i) Purity (PU) that measures how well an induced cluster corresponds to a single gold role, (ii) Collocation (CO) that measures how well a gold role corresponds to a single induced cluster, and (iii) F1 which is the harmonic mean of PU and CO. Final scores are computed by weighting each predicate by the number of its argument instances. We chose a uniform Dirichlet prior with concentration parameter as 0.1 for all the model parameters in Algorithm 1 (set roughly, without optimization[1]). 50 training iterations were used.

3.1 Results

Since the dataset has 21 semantic roles in total, we fix the total number of roles in our model to be 21. Further, we set the number of PRs to 2 (excluding $START$, END and $PRED$), and SRs to 21-2=19.

[1]Removing the Bayesian priors completely, resulted in the EM algorithm getting to a local maxima quite early, giving a substantially lower performance.

	Model	Features	PU	CO	F1
0	Baseline[2]	*d*	81.6	78.1	79.8
1a	Proposed	*d*	82.3	78.6	80.4
1b	Proposed	*d,h*	82.7	77.2	79.9
1c	Proposed	*d,p-h*	83.5	78.5	80.9
1d	Proposed	*d,p-h,h*	83.2	77.1	80.0

Table 1: Evaluation. *d* refers to *deprel*, *h* refers to *head* and *p-h* refers to *pos-head*.

# PRs	PU	CO	F1
0	81.67	78.07	79.83
1	82.91	78.99	80.90
2	83.54	78.47	80.93
3	83.68	78.23	80.87
4	83.72	78.08	80.80

Table 2: Performance variation with the number of PRs (excluding $START$, END and $PRED$)

Table 1 gives the results using different feature combinations. Line 0 reports the performance of Lang and Lapata (2011b)'s baseline, which has been shown difficult to outperform. This baseline maps 20 most frequent *deprel* to a role each, and the rest are mapped to the 21st role. By just using *deprel* as a feature, the proposed model outperforms the baseline by 0.6 points in terms of F1 score. In this configuration, the only addition over the baseline is the ordering model. Adding *head* as a feature leads to sparsity, which results in a substantial decrease in collocation (lines 1b and 1d). However, just adding *pos-head* (line 1c) does not cause this problem and gives the best F1 score. To address sparsity, we induced a distributed hidden representation for each word via a neural network, capturing the semantic similarity between words. Preliminary experiments improved the F1 score when using this word representation as a feature instead of the word directly.

Lang and Lapata (2011b) give the results of three methods on this task. In terms of F1 score, the *Latent Logistic* and *Graph Partitioning* methods result in slight reduction in performance over the baseline, while the *Split-Merge* method results in an improvement of 0.6 points. Table 1, line 1c achieves an improvement of 1.1 points over the baseline.

3.2 Further Evaluation

Table 2 shows the variation in performance w.r.t. the number of PRs[3] in the best performing configuration (Table 1, line 1c). On one extreme, when there are 0 PRs, there are only two possible intervals: $(START, PRED)$ and $(PRED, END)$ which means that the only context information a SR has is whether it is to the left or right of the predicate. With only this additional ordering information, the performance is the same as the baseline. Adding just 1 PR leads to a big increase in both purity and collocation. Increasing the number of PRs beyond 1 leads to a gradual increase in purity and decline in collocation, with the best F1 score at 2 PRs. This behavior could be explained by the fact that increasing the number of PRs also increases the number of intervals, which makes the probability distributions more sparse. In the extreme case, where all the roles are PRs and there are no SRs, the model would just learn the complete sequence of roles, which would make the parameter space too large to be tractable.

For calculating purity, each induced cluster (or role) is mapped to a particular gold role that has the maximum instances in the cluster. Analyzing the output of our model (line 1c in Table 1), we found that about 98% of the PRs and 40% of the SRs got mapped to the gold core roles ($A0$,$A1$, etc.). This suggests that the model is indeed following the intuition that (i) the ordering of core roles is important information for SRL systems, and (ii) the intervals bounded by core roles provide good context information for classification of other roles.

4 Conclusions

We propose a unified generative model for unsupervised semantic role induction that incorporates global role correlations as well as local feature information. The results indicate that a small number of ordered primary roles (PRs) is a good representation of global ordering constraints for SRL. This representation keeps the parameter space small enough for unsupervised learning.

Acknowledgments

This work was funded by the Swiss NSF grant 200021_125137 and EC FP7 grant PARLANCE.

[2]The baseline F1 reported by Lang and Lapata (2011b) is 79.5 due to a bug in their system (personal communication).

[3]Note that the system might not use all available PRs to label a given frame instance. #PRs refers to the max #PRs.

References

J.K. Baker. 1979. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America*, 65:S132.

P. Diderichsen. 1966. *Elementary Danish Grammar*. Gyldendal, Copenhagen.

E. Drach. 1937. *Grundstellung der Deutschen Satzlehre*. Diesterweg, Frankfurt.

D. Gildea and D. Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

T. Grenager and C.D. Manning. 2006. Unsupervised discovery of a statistical verb lexicon. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 1–8. Association for Computational Linguistics.

D. Klein and C.D. Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 478. Association for Computational Linguistics.

J. Lang and M. Lapata. 2011a. Unsupervised semantic role induction via split-merge clustering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Portland, Oregon.*

J. Lang and M. Lapata. 2011b. Unsupervised semantic role induction with graph partitioning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1320–1331, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

L. Màrquez, X. Carreras, K.C. Litkowski, and S. Stevenson. 2008. Semantic role labeling: an introduction to the special issue. *Computational linguistics*, 34(2):145–159.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

S. Pradhan, K. Hacioglu, V. Krugler, W. Ward, J.H. Martin, and D. Jurafsky. 2005. Support vector learning for semantic argument classification. *Machine Learning*, 60(1):11–39.

V. Punyakanok, D. Roth, W. Yih, and D. Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1346. Association for Computational Linguistics.

M. Surdeanu, R. Johansson, A. Meyers, L. Màrquez, and J. Nivre. 2008. The conll-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177. Association for Computational Linguistics.

C. Thompson, R. Levy, and C. Manning. 2003. A generative model for semantic role labeling. *Machine Learning: ECML 2003*, pages 397–408.

K. Toutanova, A. Haghighi, and C.D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2):161–191.

Humor as Circuits in Semantic Networks

Igor Labutov
Cornell University
iil4@cornell.edu

Hod Lipson
Cornell University
hod.lipson@cornell.edu

Abstract

This work presents a first step to a general implementation of the Semantic-Script Theory of Humor (SSTH). Of the scarce amount of research in computational humor, no research had focused on humor generation beyond simple puns and punning riddles. We propose an algorithm for mining simple humorous scripts from a semantic network (ConceptNet) by specifically searching for dual scripts that jointly maximize overlap and incongruity metrics in line with Raskin's Semantic-Script Theory of Humor. Initial results show that a more relaxed constraint of this form is capable of generating humor of deeper semantic content than wordplay riddles. We evaluate the said metrics through a user-assessed quality of the generated two-liners.

1 Introduction

While of significant interest in linguistics and philosophy, humor had received less attention in the computational domain. And of that work, most recent is predominately focused on humor recognition. See (Ritchie, 2001) for a good review. In this paper we focus on the problem of humor generation. While humor/sarcasm recognition merits direct application to the areas such as information retrieval (Friedland and Allan, 2008), sentiment classification (Mihalcea and Strapparava, 2006), and human-computer interaction (Nijholt et al., 2003), the application of humor generation is not any less significant. First, a good generative model of humor has the potential to outperform current discriminative models for humor recognition. Thus, ability to generate humor will potentially lead to better humor detection. Second, a computational model that conforms to the verbal theory of humor is an accessible avenue for verifying the psycholinguistic theory. In this paper we take the Semantic Script Theory of Humor (SSTH) (Attardo and Raskin, 1991) - a widely accepted theory of verbal humor and build a generative model that conforms to it.

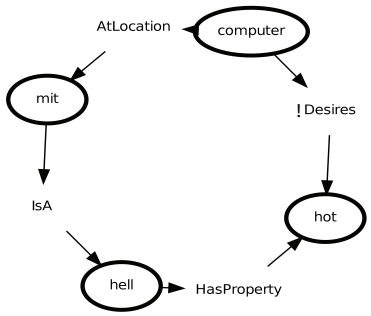


Figure 1: Semantic circuit

Much of the existing work in humor generation had focused on puns and punning riddles - humor that is centered around wordplay. And while more recent of such implementations (Hempelmann et al., 2006) take a knowledge-based approach that is rooted in the linguistic theory (SSTH), the constraint, nevertheless, significantly limits the potential of SSTH. To our knowledge, our work is the first attempt to instantiate the theory at the fundamental level, without imposing constraints on phonological similarity, or a restricted set of domain oppositions.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 150–155,
Jeju, Republic of Korea, 8-14 July 2012.

1.1 Semantic Script Theory of Humor

The Semantic Script Theory of Humor (SSTH) provides machinery to formalize the structure of most types of verbal humor (Ruch et al., 1993). SSTH posits an existence of two underlying scripts, one of which is more obvious than the other. To be humorous, the underlying scripts must satisfy two conditions: overlap and incongruity. In the setup phase of the joke, instances of the two scripts are presented in a way that does not give away the less obvious script (due to their overlap). In the punchline (resolution), a trigger expression forces the audience to switch their interpretation to the alternate (less likely) script. The alternate script must differ significantly in meaning (be incongruent with the first script) for the switch to have a humorous effect. An example below illustrates this idea (S_1 is the obvious script, and S_2 is the alternate script. Bracketed phrases are labeled with the associated script).

``Is the [doctor]$_{S_1}$ at home?''
the [patient]$_{S_1}$ asked in his
[bronchial]$_{S_1}$ [whisper]$_{S_2}$. ``No,''
the [doctor's]$_{S_1}$ [young and pretty
wife]$_{S_2}$ [whispered]$_{S_2}$ in reply.
[``Come right in.'']$_{S_2}$ (Raskin, 1985)

2 Related Work

Of the early prototypes of pun-generators, JAPE (Binsted and Ritchie, 1994), and its successor, STANDUP (Ritchie et al., 2007), produced question/answer punning riddles from general non-humorous lexicon. While humor in the generated puns could be explained by SSTH, the SSTH model itself was not employed in the process of generation. Recent work of Hempelmann (2006) comes closer to utilizing SSTH. While still focused on generating puns, they do so by explicitly defining and applying script opposition (SO) using ontological semantics. Of the more successful pun generators are systems that exploit lexical resources. HAHAcronym (Stock and Strapparava, 2002), a system for generating humorous acronyms, for example, utilizes WordNet-Domains to select phonologically similar concepts from semantically disparate domains. While the degree of humor sophistication from the above systems varies with the sophistication of the method (lexical resources, surface realizers), they all, without exception, rely on phonological constraints to produce script opposition, whereas a phonological constraint is just one of the many ways to generate script opposition.

3 System overview

ConceptNet (Liu and Singh, 2004) lends itself as an ideal ontological resource for script generation. As a network that connects everyday concepts and events with a set of causal and spatial relationships, the relational structure of ConceptNet parallels the structure of the fabula model of story generation - namely the General Transition Network (GTN) (Swartjes and Theune, 2006). As such, we hypothesize that there exist paths within the ConceptNet graph that can be represented as feasible scripts in the surface form. Moreover, multiple paths between two given nodes represent overlapping scripts - a necessary condition for verbal humor in SSTH. Given a semantic network hypergraph $G = (V, \mathcal{L})$ where $V \in Concepts$, $\mathcal{L} \in Relations$, we hypothesize that it is possible to search for script-pairs as semantic circuits that can be converted to a surface form of the Question/Answer format. We define a circuit as two paths from root *A* that terminate at a common node *B*. Our approach is composed of three stages - (1) we build a script model (SM) that captures likely transitions between concepts in a surface-realizable sequence, (2) The script model (SM) is then employed to generate a set of feasible circuits from a user-specified root node through spreading activation, producing a set of ranked scripts. (3) Ranked scripts are converted to surface form by aligning a subset of its concepts to natural language templates of the Question/Answer form. Alignment is performed through a scoring heuristic which greedily optimizes for incongruity of the surface form.

3.1 Script model

We model a script as a first order Markov chain of relations between concepts. Given a seed concept, depth-first search is performed starting from the root concept, considering all directed paths terminating at the same node as candidates for feasible script pairs. Most of the found semantic circuits, however,

do not yield a meaningful surface form and need to be pruned. Feasible circuits are learned in a supervised way, where binary labels assign each candidate circuit one of the two classes {`feasible`, `infeasible`} (we used 8 seed concepts, with 300 generated circuits for each concept). Learned transition probabilities are capable of capturing primitive stories with events, consequences, as well as appropriate qualifiers of certainty, time, size, location. Given a chain of concepts S (from hereon referred to as a script) $c_1, c_2 ... c_n$, we obtain its likelihood $\Pr(S) = \prod \Pr(r_{ij}|r_{jk})$, where r_{ij} and r_{jk} are directed relations joining concepts $< c_i, c_j >$, and $< c_j, c_k >$ respectively, and the conditionals are computed from the maximum likelihood estimate of the training data.

3.2 Semantic overlap and spreading activation

While the script model is able to capture semantically meaningful transitions in a single script, it does not capture inter-script measures such as overlap and incongruity. We employ a modified form of spreading activation with fan-out and path constraints to find semantic circuits while maximizing their semantic overlap. Activation starts at the user-specified root concept and radiates along outgoing edges. Edge pairs are weighted with their respective transition probabilities $\Pr(r_{ij}|r_{jk})$ and a decay factor $\gamma < 1$ to penalize for long scripts. An additional fan-out constraint penalizes nodes with a large number of outgoing edges (concepts that are too general to be interesting). The weight of a current node $w(c_i)$ is given by:

$$w(c_i) = \sum_{c_k \in f_{in}(c_j)} \sum_{c_j \in f_{in}(c_i)} \frac{\Pr(r_{ij}|r_{jk})}{|f_{out}(c_i)|} \gamma w(c_j) \quad (1)$$

Termination condition is satisfied when the activation weights fall below a threshold (loop checking is performed to prevent feedback). Upon termination, nodes are ranked by their activation weight, and for each node above a specified rank, a set of paths (scripts) $S_k \in \mathcal{S}$ is scored according to:.

$$\phi_k = |S_k| \log \gamma + \sum_{i}^{|S_k|} \log \Pr_k(r_{i+1}|r_i) \quad (2)$$

where ϕ_k is decay-weighted log-likelihood of script S_k in a given circuit and $|S_k|$ is the length of script

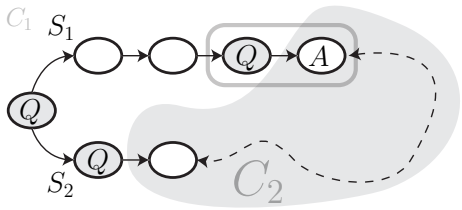


Figure 2: Question(Q) and Answer(A) concepts within the semantic circuit. Areas C_1 and C_2 represent different semantic clusters. Note that the answer(A) concept is chosen from a different cluster than the question concepts

S_k (number of nodes in the k^{th} chain). A set of scripts $\mathcal{S}$ with the highest scores in the highest ranking circuits represent scripts that are likely to be feasible and display a significant amount of semantic overlap within the circuit.

3.3 Incongruity and surface realization

The task is to select a script pair $\{S_i, S_j \, i \neq j\} \in \mathcal{S} \times \mathcal{S}$ and a set of concepts $\mathcal{C} \in S_i \cup S_j$ that will align with some surface template, while maximizing inter-script incongruity. As a measure of concept incongruity, we hierarchically cluster the entire ConceptNet using a Fast Community Detection algorithm (Clauset et al., 2004). We observe that clusters are generated for related concepts, such as *religion, marriage, computers*. Each template presents up to two concepts $\{c_1 \in S_i, c_2 \in S_j \, i \neq j\}$ in the question sentence (Q in Figure 2), and one concept $c_3 \in S_i \cup S_j$ in the answer sentence (A in Figure 2). The motivation of this approach is that the two concepts in the question are selected from two different scripts but from the same cluster, while the answer concept is selected from one of the two scripts and from a different cluster. The effect the generated two-liner produces is that of a setup and resolution (punchline), where the question intentionally sets up two parallel and compatible scripts, and the answer triggers the script switch. Below are the top-ranking two-liners as rated by a group of fifteen subjects (testing details in the next section). Each concept is indicated in brackets and labeled with the script from which the concept had originated:

`Why does the [priest]`$_{root}$ `[kneel]`$_{S_1}$ `in [church]`$_{S_2}$`? Because the [priest]`$_{root}$ `wants to [propose woman]`$_{S_1}$

Why does the [priest]$_{root}$ [drink coffee]$_{S_1}$ and [believe god]$_{S_2}$? Because the [priest]$_{root}$ wants to [wake up]$_{S_1}$

Why is the [computer]$_{root}$ [hot]$_{S_1}$ in [mit]$_{S_2}$? Because [mit]$_{S_2}$ is [hell]$_{S_2}$

Why is the [computer]$_{root}$ in [hospital]$_{S_1}$? Because the [computer]$_{root}$ has [virus]$_{S_2}$

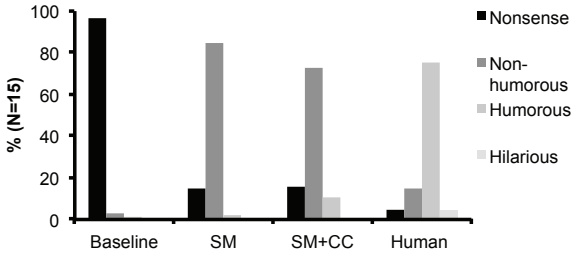


Figure 3: Human blind evaluation of generated two-liners

4 Results

We evaluate the generated two-liners by presenting them as human-generated to remove possible bias. Fifteen subjects ($N = 15$, 12 male, 3 female - graduate students in Mechanical Engineering and Computer Science departments) were presented 48 highest ranking two-liners, and were asked to rate each joke on the scale of 1 to 4 according to four categories: *hilarious (4), humorous (3), not humorous (2), nonsense(1)*. Each two-liner was generated from one of the three root categories (12 two-liners in each): *priest, woman, computer, robot*, and to normalize against individual humor biases, human-made two-liners were mixed in in the same categories. Two-liners generated by three different algorithms were evaluated by each subject:

Script model + Concept clustering (SM+CC) Both script opposition and incongruity are favored through spreading activation and concept clustering.

Script model only (SM) No concept clustering is employed. Adherence of scripts to the script model is ensured through spreading activation.

Baseline Loops are generated from a user-specified root using depth first search. Loops are pruned only to satisfy surface templates.

We compare the average scores between the two-liners generated using both the script model and concept clustering (SM+CC) (*MEAN*=1.95, *STD*=0.27) and the baseline (*MEAN*=1.06, *STD*=0.58). We observe that SM+CC algorithm yields significantly higher-scoring two-liners (one-sided t-test) with 95% confidence.

We observe that the fraction of non-humorous and nonsensical two-liners generated is still significant. Many non-humorous (but semantically sound) two-liners were formed due to erroneous labels on the concept clusters. While clustering provides a fundamental way to generate incongruity, noise in the ConceptNet often leads of cluster overfitting, and assigns related concepts into separate clusters.

Nonsensical two-liners are primarily due to the inconsistencies in POS with relation types within the ConceptNet. Because our surface form templates assume a part of speech, or a phrase type from the ConceptNet specification, erroneous entries produce nonsensical results. We partially address the problem by pruning low-scoring concepts (ConceptNet features a *SCORE* attribute reflecting the number of user votes for the concept), and all terminal nodes from consideration (nodes that are not expanded by users often indicate weak relationships).

5 Future Work

Through observation of the generated semantic paths, we note that more complex narratives, beyond questions/answer forms can be produced from the ConceptNet. Relaxing the rigid template constraint of the surface realizer will allow for more diverse types of generated humor. To mitigate the fragility of concept clustering, we are augmenting the ConceptNet with additional resources that provide domain knowledge. Resources such as SenticNet (WordNet-Affect aligned with ConceptNet) (Cambria et al., 2010b), and WordNet-Domains (Kolte and Bhirud, 2008) are both viable avenues for robust concept clustering and incongruity generation.

Acknowledgement

This paper is for my Babishan - the most important person in my life.

Huge thanks to Max Kelner - those everyday teas at Mattins and continuous inspiration.

This work was supported in part by NSF CDI Grant ECCS 0941561. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the sponsoring organizations.

References

S. Attardo and V. Raskin. 1991. Script theory revis (it) ed: Joke similarity and joke representation model. *Humor: International Journal of Humor Research; Humor: International Journal of Humor Research*.

K. Binsted and G. Ritchie. 1994. A symbolic description of punning riddles and its computer implementation. *Arxiv preprint cmp-lg/9406021*.

K. Binsted, A. Nijholt, O. Stock, C. Strapparava, G. Ritchie, R. Manurung, H. Pain, A. Waller, and D. O'Mara. 2006. Computational humor. *Intelligent Systems, IEEE*, 21(2):59–69.

K. Binsted. 1996. Machine humour: An implemented model of puns.

E. Cambria, A. Hussain, C. Havasi, and C. Eckl. 2010a. Senticspace: visualizing opinions and sentiments in a multi-dimensional vector space. *Knowledge-Based and Intelligent Information and Engineering Systems*, pages 385–393.

E. Cambria, R. Speer, C. Havasi, and A. Hussain. 2010b. Senticnet: A publicly available semantic resource for opinion mining. In *Proceedings of the 2010 AAAI Fall Symposium Series on Commonsense Knowledge*.

A. Clauset, M.E.J. Newman, and C. Moore. 2004. Finding community structure in very large networks. *Physical review E*, 70(6):066111.

F. Crestani. 1997. Retrieving documents by constrained spreading activation on automatically constructed hypertexts. In *EUFIT 97-5th European Congress on Intelligent Techniques and Soft Computing. Germany*. Citeseer.

L. Friedland and J. Allan. 2008. Joke retrieval: recognizing the same joke told differently. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 883–892. ACM.

C.F. Hempelmann, V. Raskin, and K.E. Triezenberg. 2006. Computer, tell me a joke... but please make it funny: Computational humor with ontological semantics. In *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference, Melbourne Beach, Florida, USA, May 11*, volume 13, pages 746–751.

S.G. Kolte and S.G. Bhirud. 2008. Word sense disambiguation using wordnet domains. In *Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on*, pages 1187–1191. IEEE.

H. Liu and P. Singh. 2004. Conceptneta practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226.

R. Mihalcea and C. Strapparava. 2006. Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence*, 22(2):126–142.

M.E.J. Newman. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582.

A. Nijholt, O. Stock, A. Dix, and J. Morkes. 2003. Humor modeling in the interface. In *CHI'03 extended abstracts on Human factors in computing systems*, pages 1050–1051. ACM.

V. Raskin. 1998. The sense of humor and the truth. *The Sense of Humor. Explorations of a Personality Characteristic, Berlin: Mouton De Gruyter*, pages 95–108.

G. Ritchie, R. Manurung, H. Pain, A. Waller, R. Black, and D. OMara. 2007. A practical application of computational humour. In *Proceedings of the 4th. International Joint Workshop on Computational Creativity, London, UK*.

G. Ritchie. 2001. Current directions in computational humour. *Artificial Intelligence Review*, 16(2):119–135.

W. Ruch, S. Attardo, and V. Raskin. 1993. Toward an empirical verification of the general theory of verbal humor. *Humor: International Journal of Humor Research; Humor: International Journal of Humor Research*.

J. Savoy. 1992. Bayesian inference networks and spreading activation in hypertext systems. *Information processing & management*, 28(3):389–406.

S. Spagnola and C. Lagoze. 2011. Edge dependent pathway scoring for calculating semantic similarity in conceptnet. In *Proceedings of the Ninth International Conference on Computational Semantics*, pages 385–389. Association for Computational Linguistics.

O. Stock and C. Strapparava. 2002. Hahacronym: Humorous agents for humorous acronyms. *Stock, Oliviero, Carlo Strapparava, and Anton Nijholt. Eds*, pages 125–135.

I. Swartjes and M. Theune. 2006. A fabula model for emergent narrative. *Technologies for Interactive Digital Storytelling and Entertainment*, pages 49–60.

J.M. Taylor and L.J. Mazlack. 2004. Humorous wordplay recognition. In *Systems, Man and Cybernetics, 2004 IEEE International Conference on*, volume 4, pages 3306–3311. IEEE.

J. Taylor and L. Mazlack. 2005. Toward computational recognition of humorous intent. In *Proceedings of Cognitive Science Conference*, pages 2166–2171.

J.M. Taylor. 2009. Computational detection of humor: A dream or a nightmare? the ontological semantics approach. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 03*, pages 429–432. IEEE Computer Society.

Crowdsourcing Inference-Rule Evaluation

Naomi Zeichner
Bar-Ilan University
Ramat-Gan, Israel
zeichner.naomi@gmail.com

Jonathan Berant
Tel-Aviv University
Tel-Aviv, Israel
jonatha6@post.tau.ac.il

Ido Dagan
Bar-Ilan University
Ramat-Gan, Israel
dagan@cs.biu.ac.il

Abstract

The importance of inference rules to semantic applications has long been recognized and extensive work has been carried out to automatically acquire inference-rule resources. However, evaluating such resources has turned out to be a non-trivial task, slowing progress in the field. In this paper, we suggest a framework for evaluating inference-rule resources. Our framework simplifies a previously proposed "instance-based evaluation" method that involved substantial annotator training, making it suitable for crowdsourcing. We show that our method produces a large amount of annotations with high inter-annotator agreement for a low cost at a short period of time, without requiring training expert annotators.

1 Introduction

Inference rules are an important component in semantic applications, such as Question Answering (QA) (Ravichandran and Hovy, 2002) and Information Extraction (IE) (Shinyama and Sekine, 2006), describing a directional inference relation between two text patterns with variables. For example, to answer the question *'Where was Reagan raised?'* a QA system can use the rule *'X brought up in Y→X raised in Y'* to extract the answer from *'Reagan was brought up in Dixon'*. Similarly, an IE system can use the rule *'X work as Y→X hired as Y'* to extract the PERSON and ROLE entities in the "hiring" event from *'Bob worked as an analyst for Dell'*.

The significance of inference rules has led to substantial effort into developing algorithms that automatically learn inference rules (Lin and Pantel, 2001; Sekine, 2005; Schoenmackers et al., 2010), and generate knowledge resources for inference systems. However, despite their potential, utilization of inference rule resources is currently somewhat limited. This is largely due to the fact that these algorithms often produce invalid rules. Thus, evaluation is necessary both for resource developers as well as for inference system developers who want to asses the quality of each resource. Unfortunately, as evaluating inference rules is hard and costly, there is no clear evaluation standard, and this has become a slowing factor for progress in the field.

One option for evaluating inference rule resources is to measure their impact on an end task, as that is what ultimately interests an inference system developer. However, this is often problematic since inference systems have many components that address multiple phenomena, and thus it is hard to assess the effect of a single resource. An example is the Recognizing Textual Entailment (RTE) framework (Dagan et al., 2009), in which given a text T and a textual hypothesis H, a system determines whether H can be inferred from T. This type of evaluation was established in RTE challenges by ablation tests (see RTE ablation tests in ACLWiki) and showed that resources' impact can vary considerably from one system to another. These issues have also been noted by Sammons et al. (2010) and LoBue and Yates (2011). A complementary application-independent evaluation method is hence necessary.

Some attempts were made to let annotators judge rule correctness *directly*, that is by asking them to judge the correctness of a given rule (Shinyama et al., 2002; Sekine, 2005). However, Szpektor et al. (2007) observed that directly judging rules out of context often results in low inter-annotator agreement. To remedy that, Szpektor et al. (2007) and

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 156–160,
Jeju, Republic of Korea, 8-14 July 2012.

Bhagat et al. (2007) proposed *"instance-based evaluation"*, in which annotators are presented with an *application* of a rule in a particular context and need to judge whether it results in a valid inference. This simulates the utility of rules in an application and yields high inter-annotator agreement. Unfortunately, their method requires lengthy guidelines and substantial annotator training effort, which are time consuming and costly. Thus, a simple, robust and replicable evaluation method is needed.

Recently, crowdsourcing services such as Amazon Mechanical Turk (AMT) and CrowdFlower (CF)[1] have been employed for semantic inference annotation (Snow et al., 2008; Wang and Callison-Burch, 2010; Mehdad et al., 2010; Negri et al., 2011). These works focused on generating and annotating RTE text-hypothesis pairs, but did not address annotation and evaluation of inference rules. In this paper, we propose a novel instance-based evaluation framework for inference rules that takes advantage of crowdsourcing. Our method substantially simplifies annotation of rule applications and avoids annotator training completely. The novelty in our framework is two-fold: (1) We simplify instance-based evaluation from a complex decision scenario to two independent binary decisions. (2) We apply methodological principles that efficiently communicate the definition of the "inference" relation to untrained crowdsourcing workers (*Turkers*).

As a case study, we applied our method to evaluate algorithms for learning inference rules between predicates. We show that we can produce many annotations cheaply, quickly, at good quality, while achieving high inter-annotator agreement.

2 Evaluating Rule Applications

As mentioned, in instance-based evaluation individual rule applications are judged rather than rules in isolation, and the quality of a rule-resource is then evaluated by the validity of a sample of applications of its rules. Rule application is performed by finding an instantiation of the rule *left-hand-side* in a corpus (termed *LHS extraction*) and then applying the rule on the extraction to produce an instantiation of the rule *right-hand-side* (termed *RHS instantiation*). For example, the rule *'X observe Y→X celebrate Y'* can be applied on the LHS extraction *'they observe holidays'* to produce the RHS instantiation *'they celebrate holidays'*.

The target of evaluation is to judge whether each rule application is valid or not. Following the standard RTE task definition, a rule application is considered valid if a human reading the *LHS extraction* is highly likely to infer that the *RHS instantiation* is true (Dagan et al., 2009). In the aforementioned example, the annotator is expected to judge that *'they observe holidays'* entails *'they celebrate holidays'*. In addition to this straightforward case, two more subtle situations may arise. The first is that the LHS extraction is meaningless. We regard a proposition as meaningful if a human can easily understand its meaning (despite some simple grammatical errors). A meaningless LHS extraction usually occurs due to a faulty extraction process (e.g., Table 1, Example 2) and was relatively rare in our case study (4% of output, see Section 4). Such rule applications can either be extracted from the sample so that the rule-base is not penalized (since the problem is in the extraction procedure), or can be used as examples of non-entailment, if we are interested in overall performance. A second situation is a meaningless RHS instantiation, usually caused by rule application in a wrong context. This case is tagged as non-entailment (for example, applying the rule *'X observe Y→X celebrate Y'* in the context of the extraction *'companies observe dress code'*).

Each rule application therefore requires an answer to the following three questions: 1) Is the LHS extraction meaningful? 2) Is the RHS instantiation meaningful? 3) If both are meaningful, does the LHS extraction entail the RHS instantiation?

3 Crowdsourcing

Previous works using crowdsourcing noted some principles to help get the most out of the service(Wang et al., 2012). In keeping with these findings we employ the following principles: **(a) Simple tasks**. The global task is split into simple sub-tasks, each dealing with a single aspect of the problem. **(b) Do not assume linguistic knowledge by annotators**. Task descriptions avoid linguistic terms such as "tense", which confuse workers. **(c) Gold standard validation**. Using CF's built-in methodology,

[1] https://www.mturk.com and http://crowdflower.com

Phrase	Meaningful	Comments
1) Doctors be treat Mary	Yes	Annotators are instructed to ignore simple inflectional errors
2) A player deposit an	No	Bad extraction for the rule LHS *'X deposit Y'*
3) humans bring in bed	No	Wrong context, result of applying *'X turn in Y→X bring in Y'* on *'humans turn in bed'*

Table 1: Examples of phrase "meaningfulness" (Note that the comments are not presented to Turkers).

gold standard (GS) examples are combined with actual annotations to continuously validate annotator reliability.

We split the annotation process into two tasks, the first to judge phrase meaningfulness (Questions 1 and 2 above) and the second to judge entailment (Question 3 above). In Task 1, the LHS extractions and RHS instantiations of all rule applications are separated and presented to different Turkers independently of one another. This task is simple, quick and cheap and allows Turkers to focus on the single aspect of judging phrase meaningfulness. Rule applications for which both the LHS extraction and RHS instantiation are judged as meaningful are passed to Task 2, where Turkers need to decide whether a given rule application is valid. If not for Task 1, Turkers would need to distinguish in Task 2 between non-entailment due to (1) an incorrect rule (2) a meaningless RHS instantiation (3) a meaningless LHS extraction. Thanks to Task 1, Turkers are presented in Task 2 with two meaningful phrases and need to decide only whether one entails the other.

To ensure high quality output, each example is evaluated by three Turkers. Similarly to Mehdad et al. (2010) we only use results for which the confidence value provided by CF is greater than 70%.

We now describe the details of both tasks. Our simplification contrasts with Szpektor et al. (2007), whose judgments for each rule application are similar to ours, but had to be performed simultaneously by annotators, which required substantial training.

Task 1: *Is the phrase meaningful?*
In keeping with the second principle above, the task description is made up of a short verbal explanation followed by positive and negative examples. The definition of "meaningfulness" is conveyed via examples pointing to properties of the automatic phrase extraction process, as seen in Table 1.

Task 2: *Judge if one phrase is true given another.*
As mentioned, rule applications for which both sides were judged as meaningful are evaluated for entailment. The challenge is to communicate the definition of "entailment" to Turkers. To that end the task description begins with a short explanation followed by "easy" and "hard" examples with explanations, covering a variety of positive and negative entailment "types" (Table 2).

Defining "entailment" is quite difficult when dealing with expert annotators and still more with non-experts, as was noted by Negri et al. (2011). We therefore employ several additional mechanisms to get the definition of entailment across to Turkers and increase agreement with the GS. We run an initial small test run and use its output to improve annotation in two ways: First, we take examples that were "confusing" for Turkers and add them to the GS with explanatory feedback presented when a Turker answers incorrectly. (E.g., the pair *('The owner be happy to help drivers', 'The owner assist drivers')* was judged as entailing in the test run but only achieved a confidence value of 0.53). Second, we add examples that were annotated unanimously by Turkers to the GS to increase its size, allowing CF to better estimate Turker's reliability (following CF recommendations, we aim to have around 10% GS examples in every run). In Section 4 we show that these mechanisms improved annotation quality.

4 Case Study

As a case study, we used our evaluation methodology to compare four methods for learning entailment rules between predicates: DIRT (Lin and Pantel, 2001), Cover (Weeds and Weir, 2003), BInc (Szpektor and Dagan, 2008) and Berant et al. (2010). To that end, we applied the methods on a set of one billion extractions (generously provided by Fader et al. (2011)) automatically extracted from the ClueWeb09 web crawl[2], where each extraction comprises a predicate and two arguments. This resulted in four learned inference rule resources.

[2]http://lemurproject.org/clueweb09.php/

Example	Entailed	Explanation given to Turkers
LHS: The lawyer sign the contract RHS: The lawyer read the contract	Yes	There is a chance the lawyer has not read the contract, but most likely that as he signed it, he must have read it.
LHS: John be related to Jerry RHS: John be a close relative of Jerry	No	The LHS can be understood from the RHS, but not the other way around as the LHS is more general.
LHS: Women be at increased risk of cancer RHS: Women die of cancer	No	Although the RHS is correct, it cannot be understood from the LHS.

Table 2: Examples given in the description of Task 2.

We randomly sampled 5,000 extractions, and for each one sampled four rules whose LHS matches the extraction from the union of the learned resources. We then applied the rules, which resulted in 20,000 rule applications. We annotated rule applications using our methodology and evaluated each learning method by comparing the rules learned by each method with the annotation generated by CF.

In Task 1, 281 rule applications were annotated as meaningless LHS extraction, and 1,012 were annotated as meaningful LHS extraction but meaningless RHS instantiation and so automatically annotated as non-entailment. 8,264 rule applications were passed on to Task 2, as both sides were judged meaningful (the remaining 10,443 discarded due to low CF confidence). In Task 2, 5,555 rule applications were judged with a high confidence and supplied as output, 2,447 of them as positive entailment and 3,108 as negative. Overall, 6,567 rule applications (dataset of this paper) were annotated for a total cost of $1000. The annotation process took about one week.

In tests run during development we experimented with Task 2 wording and GS examples, seeking to make the definition of entailment as clear as possible. To do so we randomly sampled and manually annotated 200 rule applications (from the initial 20,000), and had Turkers judge them. In our initial test, Turkers tended to answer "yes" comparing to our own annotation, with 0.79 agreement between their annotation and ours, corresponding to a kappa score of 0.54. After applying the mechanisms described in Section 3, false-positive rate was reduced from 18% to 6% while false-negative rate only increased from 4% to 5%, corresponding to a high agreement of 0.9 and kappa of 0.79.

In our test, 63% of the 200 rule applications were annotated unanimously by the Turkers. Importantly, all these examples were in perfect agreement with our own annotation, reflecting their high reliability. For the purpose of evaluating the resources learned by the algorithms we used annotations with CF confidence ≥ 0.7 for which kappa is 0.99.

Lastly, we computed the area under the recall-precision curve (AUC) for *DIRT*, *Cover*, *BInc* and *Berant et al.*'s method, resulting in an AUC of 0.4, 0.43, 0.44, and 0.52 respectively. We used the AUC curve, with number of recall-precision points in the order of thousands, to avoid tuning a threshold parameter. Overall, we demonstrated that our evaluation framework allowed us to compare four different learning methods in low costs and within one week.

5 Discussion

In this paper we have suggested a crowdsourcing framework for evaluating inference rules. We have shown that by simplifying the previously-proposed instance-based evaluation framework we are able to take advantage of crowdsourcing services to replace trained expert annotators, resulting in good quality large scale annotations, for reasonable time and cost. We have presented the methodological principles we developed to get the entailment decision across to Turkers, achieving very high agreement both with our annotations and between the annotators themselves. Using the CrowdFlower forms we provide with this paper, the proposed methodology can be beneficial for both resource developers evaluating their output as well as inference system developers wanting to assess the quality of existing resources.

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 Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 287923 (EXCITEMENT).

References

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2010. Global learning of focused entailment graphs. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*.

Rahul Bhagat, Patrick Pantel, and Eduard Hovy. 2007. LEDIR: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2009. Recognizing textual entailment: Rational, evaluation and approaches. *Natural Language Engineering*, 15(Special Issue 04):i–xvii.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing (EMNLP '11)*.

Dekang Lin and Patrick Pantel. 2001. DIRT - discovery of inference rules from text. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Peter LoBue and Alexander Yates. 2011. Types of common-sense knowledge needed for recognizing textual entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

Yashar Mehdad, Matteo Negri, and Marcello Federico. 2010. Towards cross-lingual textual entailment. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (ACL)*.

Matteo Negri, Luisa Bentivogli, Yashar Mehdad, Danilo Giampiccolo, and Alessandro Marchetti. 2011. Divide and conquer: Crowdsourcing the creation of cross-lingual textual entailment corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*.

Mark Sammons, V. G. Vinod Vydiswaran, and Dan Roth. 2010. "ask not what textual entailment can do for you...". In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*.

Stefan Schoenmackers, Oren Etzioni Jesse Davis, and Daniel S. Weld. 2010. Learning first-order horn clauses from web text. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP '10)*.

Satoshi Sekine. 2005. Automatic paraphrase discovery based on context and keywords between ne pairs. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL '06)*.

Yusuke Shinyama, Satoshi Sekine, and Kiyoshi Sudo. 2002. Automatic paraphrase acquisition from news articles. In *Proceedings of the second international conference on Human Language Technology Research (HLT '02)*.

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)*.

Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*.

Idan Szpektor, Eyal Shnarch, and Ido Dagan. 2007. Instance-based evaluation of entailment rule acquisition. In *Proceedings of the annual meeting of the Association for Computational Linguistics (ACL)*.

Rui Wang and Chris Callison-Burch. 2010. Cheap facts and counter-facts. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*.

Aobo Wang, Cong Duy Vu Hoang, and Min-Yen Kan. 2012. Perspectives on crowdsourcing annotations for natural language processing. *Journal of Language Resources and Evaluation)*.

Julie Weeds and David Weir. 2003. A general framework for distributional similarity. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2003)*.

A Comprehensive Gold Standard for the Enron Organizational Hierarchy

Apoorv Agarwal[1]* **Adinoyi Omuya[1]**** **Aaron Harnly[2]†** **Owen Rambow[3]‡**

[1] Department of Computer Science, Columbia University, New York, NY, USA
[2] Wireless Generation Inc., Brooklyn, NY, USA
[3] Center for Computational Learning Systems, Columbia University, New York, NY, USA
* apoorv@cs.columbia.edu ** awo2108@columbia.edu
†aaron@cs.columbia.edu ‡rambow@ccls.columbia.edu

Abstract

Many researchers have attempted to predict the Enron corporate hierarchy from the data. This work, however, has been hampered by a lack of data. We present a new, large, and freely available gold-standard hierarchy. Using our new gold standard, we show that a simple lower bound for social network-based systems outperforms an upper bound on the approach taken by current NLP systems.

1 Introduction

Since the release of the Enron email corpus, many researchers have attempted to predict the Enron corporate hierarchy from the email data. This work, however, has been hampered by a lack of data about the organizational hierarchy. Most researchers have used the job titles assembled by (Shetty and Adibi, 2004), and then have attempted to predict the relative ranking of two people's job titles (Rowe et al., 2007; Palus et al., 2011). A major limitation of the list compiled by Shetty and Adibi (2004) is that it only covers those "core" employees for whom the complete email inboxes are available in the Enron dataset. However, it is also interesting to determine whether we can predict the hierarchy of other employees, for whom we only have an incomplete set of emails (those that they sent to or received from the core employees). This is difficult in particular because there are dominance relations between two employees such that no email between them is available in the Enron data set. The difficulties with the existing data have meant that researchers have either not performed quantitative analyses (Rowe et al., 2007), or have performed them on very small sets: for example, (Bramsen et al., 2011a) use 142 dominance pairs for training and testing.

We present a new resource (Section 3). It is a large gold-standard hierarchy, which we extracted manually from pdf files. Our gold standard contains 1,518 employees, and 13,724 dominance pairs (pairs of employees such that the first dominates the second in the hierarchy, not necessarily immediately). All of the employees in the hierarchy are email correspondents on the Enron email database, though obviously many are not from the core group of about 158 Enron employees for which we have the complete inbox. The hierarchy is linked to a threaded representation of the Enron corpus using shared IDs for the employees who are participants in the email conversation. The resource is available as a MongoDB database.

We show the usefulness of this resource by investigating a simple predictor for hierarchy based on social network analysis (SNA), namely degree centrality of the social network induced by the email correspondence (Section 4). We call this a lower bound for SNA-based systems because we are only using a single simple metric (degree centrality) to establish dominance. Degree centrality is one of the features used by Rowe et al. (2007), but they did not perform a quantitative evaluation, and to our knowledge there are no published experiments using only degree centrality. Current systems using natural language processing (NLP) are restricted to making informed predictions on dominance pairs for which email exchange is available. We show (Section 5) that the upper bound performance of such

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 161–165,
Jeju, Republic of Korea, 8-14 July 2012.

NLP-based systems is much lower than our SNA-based system on the entire gold standard. We also contrast the simple SN-based system with a specific NLP system based on (Gilbert, 2012), and show that even if we restrict ourselves to pairs for which email exchange is available, our simple SNA-based systems outperforms the NLP-based system.

2 Work on Enron Hierarchy Prediction

The Enron email corpus was introduced by Klimt and Yang (2004). Since then numerous researchers have analyzed the network formed by connecting people with email exchange links (Diesner et al., 2005; Shetty and Adibi, 2004; Namata et al., 2007; Rowe et al., 2007; Diehl et al., 2007; Creamer et al., 2009). Rowe et al. (2007) use the email exchange network (and other features) to predict the dominance relations between people in the Enron email corpus. They however do not present a quantitative evaluation.

Bramsen et al. (2011b) and Gilbert (2012) present NLP based models to predict dominance relations between Enron employees. Neither the test-set nor the system of Bramsen et al. (2011b) is publicly available. Therefore, we compare our baseline SNA based system with that of Gilbert (2012). Gilbert (2012) produce training and test data as follows: an email message is labeled *upward* only when every recipient outranks the sender. An email message is labeled *not-upward* only when every recipient does not outrank the sender. They use an n-gram based model with Support Vector Machines (SVM) to predict if an email is of class *upward* or *not-upward*. They make the phrases (n-grams) used by their best performing system publicly available. We use their n-grams with SVM to predict dominance relations of employees in our gold standard and show that a simple SNA based approach outperforms this baseline. Moreover, Gilbert (2012) exploit dominance relations of only 132 people in the Enron corpus for creating their training and test data. Our gold standard has dominance relations for 1518 Enron employees.

3 The Enron Hierarchy Gold Standard

Klimt and Yang (2004) introduced the Enron email corpus. They reported a total of 619,446 emails taken from folders of 158 employees of the Enron corporation. We created a database of organizational hierarchy relations by studying the original Enron organizational charts. We discovered these charts by performing a manual, random survey of a few hundred emails, looking for explicit indications of hierarchy. We found a few documents with organizational charts, which were always either Excel or Visio files. We then searched all remaining emails for attachments of the same filetype, and exhaustively examined those with additional org charts. We then manually transcribed the information contained in all org charts we found.

Our resulting gold standard has a total of 1518 nodes (employees) which are described as being in immediate dominance relations (manager-subordinate). There are 2155 immediate dominance relations spread over 65 levels of dominance (CEO, manager, trader etc.) From these relations, we formed the transitive closure and obtained 13,724 hierarchal relations. For example, if A immediately dominates B and B immediately dominates C, then the set of valid organizational dominance relations are A dominates B, B dominates C and A dominates C. This data set is much larger than any other data set used in the literature for the sake of predicting organizational hierarchy.

We link this representation of the hierarchy to the threaded Enron corpus created by Yeh and Harnley (2006). They pre-processed the dataset by combining emails into threads and restoring some missing emails from their quoted form in other emails. They also co-referenced multiple email addresses belonging to one person, and assigned unique identifiers and names to persons. Therefore, each person is a-priori associated with a set of email addresses and names (or name variants), but has only one unique identifier. Our corpus contains 279,844 email messages. These messages belong to 93,421 unique persons. We use these unique identifiers to express our gold hierarchy. This means that we can easily retrieve all emails associated with people in our gold hierarchy, and we can easily determine the hierarchical relation between the sender and receivers of any email.

The whole set of person nodes is divided into two parts: **core** and **non-core**. The set of core people are those whose inboxes were taken to create the Enron

email network (a set of 158 people). The set of non-core people are the remaining people in the network who either send an email to and/or receive an email from a member of the core group. As expected, the email exchange network (the network induced from the emails) is densest among core people (density of 20.997% in the email exchange network), and much less dense among the non-core people (density of 0.008%).

Our data base is freely available as a MongoDB database, which can easily be interfaced with using APIs in various programming languages. For information about how to obtain the database, please contact the authors.

4 A Hierarchy Predictor Based on the Social Network

We construct the **email exchange network** as follows. This network is represented as an undirected weighted graph. The nodes are all the unique employees. We add a link between two employees if one sends at least one email to the other (who can be a TO, CC, or BCC recipient). The weight is the number of emails exchanged between the two. Our email exchange network consists of 407,095 weighted links and 93,421 nodes.

Our algorithm for predicting the dominance relation using social network analysis metric is simple. We calculate the degree centrality of every node in the email exchange network, and then rank the nodes by their degree centrality. Recall that the degree centrality is the proportion of nodes in the network with which a node is connected. (We also tried eigenvalue centrality, but this performed worse. For a discussion of the use of degree centrality as a valid indication of importance of nodes in a network, see (Chuah and Coman, 2009).) Let $C_D(n)$ be the degree centrality of node n, and let DOM be the dominance relation (transitive, not symmetric) induced by the organizational hierarchy. We then simply assume that for two people p_1 and p_2, if $C_D(p_1) > C_D(p_2)$, then DOM(p_1,p_2). For every pair of people who are related with an organizational dominance relation in the gold standard, we then predict which person dominates the other. Note that we do not predict if two people are in a dominance relation to begin with. The task of predicting if two people are in a dominance relation is different and we do not address that task in this paper. Therefore, we restrict our evaluation to pairs of people (p_1, p_2) who are related hierarchically (i.e., either DOM(p_1,p_2) or DOM(p_2,p_1) in the gold standard). Since we only predict the directionality of the dominance relation of people given they are in a hierarchical relation,[1] the random baseline for our task performs at 50%. We have 13,724 such pairs of people in the gold standard. When we use the network induced simply by the email exchanges, we get a remarkably high accuracy of 83.88% (Table 1). We denote this system by SNA_G.

Type	# pairs	%Acc
All	13,724	83.88
Core	440	79.31
Inter	6436	93.75
Non-Core	6847	74.57

Table 1: Prediction accuracy by type of predicted organizational dominance pair; "Inter" means that one element of the pair is from the core and the other is not; a negative error reduction indicates an increase in error

In this paper, we also make an observation crucial for the task of hierarchy prediction, based on the distinction between the core and the non-core groups (see Section 3). This distinction is crucial for this task since by definition the degree centrality measure (which depends on how accurately the underlying network expresses the communication network) suffers from missing email messages (for the non-core group). Our results in table 1 confirm this intuition. Since we have a richer network for the core group, degree centrality is a better predictor for this group than for the non-core group.

We also note that the prediction accuracy is by far the highest for the **inter** hierarchal pairs. The inter hierarchal pairs are those in which one node is from the core group of people and the other node is from the non-core group of people. This is explained by the fact that the core group was chosen by law enforcement because they were most likely to contain information relevant to the legal proceedings against Enron; i.e., the owners of the mailboxes

[1] This style of evaluation is common (Diehl et al., 2007; Bramsen et al., 2011b).

were more likely more highly placed in the hierarchy. Furthermore, because of the network characteristics described above (a relatively dense network), the core people are also more likely to have a high centrality degree, as compared to the non-core people. Therefore, the correlation between centrality degree and hierarchal dominance will be high.

5 Using NLP and SNA

In this section we compare and contrast the performance of NLP-based systems with that of SNA-based systems on the Enron hierarchy gold standard we introduce in this paper. This gold standard allows us to notice an important limitation of the NLP-based systems (for this task) in comparison to SNA-based systems in that the NLP-based systems require communication links between people to make a prediction about their dominance relation, whereas an SNA-based system may predict dominance relations without this requirement.

Table 2 presents the results for four experiments. We first determine an upper bound for current NLP-based systems. Current NLP-based systems predict dominance relations between a pair of people by using the language used in email exchanges between these people; if there is no email exchange, such methods cannot make a prediction. Let G be the set of all dominance relations in the gold standard ($|G| = 13,723$). We define $T \subset G$ to be the set of pairs in the gold standard such that the people involved in the pair in T communicate with each other. These are precisely the dominance relations in the gold standard which can be established using a current NLP-based approach. The number of such pairs is $|T| = 2,640$. Therefore, if we consider a perfect NLP system that correctly predicts the dominance of $2,640$ tuples and randomly guesses the dominance relation of the remaining $11,084$ tuples, the system would achieve an accuracy of $(2640 + 11084/2)/13724 = 59.61\%$. We refer to this number as the upper bound on the best performing NLP system for the gold standard. This upper bound of 59.61% for an NLP-based system is lower (24.27% absolute) than a simple SNA-based system (SNA_G, explained in section 4) that predicts the dominance relation for all the tuples in the gold standard G.

As explained in section 2, we use the phrases provided by Gilbert (2012) to build an NLP-based model for predicting dominance relations of tuples in set $T \subset G$. Note that we only use the tuples from the gold standard where the NLP-based system may hope to make a prediction (i.e. people in the tuple communicate via email). This system, NLP_{Gilbert} achieves an accuracy of 82.37% compared to the social network-based approach (SNA_T) which achieves a higher accuracy of 87.58% on the same test set T. This comparison shows that SNA-based approach out-performs the NLP-based approach even if we evaluate on a much smaller part of the gold standard, namely the part where an NLP-based approach does not suffer from having to make a random prediction for nodes that do not comunicate via email.

System	Test set	# test points	%Acc
UB_{NLP}	G	13,724	59.61
NLP_{Gilbert}	T	2604	82.37
SNA_T	T	2604	87.58
SNA_G	G	13,724	83.88

Table 2: Results of four systems, essentially comparing performance of purely NLP-based systems with simple SNA-based systems.

6 Future Work

One key challenge of the problem of predicting domination relations of Enron employees based on their emails is that the underlying network is incomplete. We hypothesize that SNA-based approaches are sensitive to the *goodness* with which the underlying network represents the true social network. Part of the missing network may be recoverable by analyzing the content of emails. Using sophisticated NLP techniques, we may be able to *enrich* the network and use standard SNA metrics to predict the dominance relations in the gold standard.

Acknowledgments

We would like to thank three anonymous reviewers for useful comments. This work is supported by NSF grant IIS-0713548. Harnly was at Columbia University while he contributed to the work.

References

Philip Bramsen, Martha Escobar-Molano, Ami Patel, and Rafael Alonso. 2011a. Extracting social power relationships from natural language. In *ACL*, pages 773–782. The Association for Computer Linguistics.

Philip Bramsen, Martha Escobar-Molano, Ami Patel, and Rafael Alonso. 2011b. Extracting social power relationships from natural language. *ACL*.

Mooi-Choo Chuah and Alexandra Coman. 2009. Identifying connectors and communities: Understanding their impacts on the performance of a dtn publish/subscribe system. *International Conference on Computational Science and Engineering (CSE '09)*.

Germán Creamer, Ryan Rowe, Shlomo Hershkop, and Salvatore J. Stolfo. 2009. Segmentation and automated social hierarchy detection through email network analysis. In Haizheng Zhang, Myra Spiliopoulou, Bamshad Mobasher, C. Lee Giles, Andrew Mccallum, Olfa Nasraoui, Jaideep Srivastava, and John Yen, editors, *Advances in Web Mining and Web Usage Analysis*, pages 40–58. Springer-Verlag, Berlin, Heidelberg.

Christopher Diehl, Galileo Mark Namata, and Lise Getoor. 2007. Relationship identification for social network discovery. *AAAI '07: Proceedings of the 22nd National Conference on Artificial Intelligence*.

Jana Diesner, Terrill L Frantz, and Kathleen M Carley. 2005. Communication networks from the enron email corpus it's always about the people. enron is no different. *Computational & Mathematical Organization Theory*, 11(3):201–228.

Eric Gilbert. 2012. Phrases that signal workplace hierarchy. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work (CSCW)*.

Bryan Klimt and Yiming Yang. 2004. Introducing the enron corpus. *In First Conference on Email and Anti-Spam (CEAS)*.

Galileo Mark S. Namata, Jr., Lise Getoor, and Christopher P. Diehl. 2007. Inferring organizational titles in online communication. In *Proceedings of the 2006 conference on Statistical network analysis*, ICML'06, pages 179–181, Berlin, Heidelberg. Springer-Verlag.

Sebastian Palus, Piotr Brodka, and Przemysław Kazienko. 2011. Evaluation of organization structure based on email interactions. *International Journal of Knowledge Society Research*.

Ryan Rowe, German Creamer, Shlomo Hershkop, and Salvatore J Stolfo. 2007. Automated social hierarchy detection through email network analysis. *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 109–117.

Jitesh Shetty and Jaffar Adibi. 2004. Ex employee status report. `http://www.isi.edu/~adibi/Enron/Enron_Employee_Status.xls`.

Jen Yuan Yeh and Aaron Harnley. 2006. Email thread reassembly using similarity matching. *In Proceedings of CEAS*.

A Two-step Approach to Sentence Compression of Spoken Utterances

Dong Wang, Xian Qian, Yang Liu
The University of Texas at Dallas
dongwang,qx,yangl@hlt.utdallas.edu

Abstract

This paper presents a two-step approach to compress spontaneous spoken utterances. In the first step, we use a sequence labeling method to determine if a word in the utterance can be removed, and generate n-best compressed sentences. In the second step, we use a discriminative training approach to capture sentence level global information from the candidates and rerank them. For evaluation, we compare our system output with multiple human references. Our results show that the new features we introduced in the first compression step improve performance upon the previous work on the same data set, and reranking is able to yield additional gain, especially when training is performed to take into account multiple references.

1 Introduction

Sentence compression aims to preserve the most important information in the original sentence with fewer words. It can be used for abstractive summarization where extracted important sentences often need to be compressed and merged. For summarization of spontaneous speech, sentence compression is especially important, since unlike fluent and well-structured written text, spontaneous speech contains a lot of disfluencies and much redundancy. The following shows an example of a pair of source and compressed spoken sentences[1] from human annotation (removed words shown in bold):

[**original sentence**]
and then **um** *in terms of the source* **the things uh** *the only things that we had on there* **I believe** *were whether...*

[**compressed sentence**]
and then in terms of the source the only things that we had on there were whether...

In this study we investigate sentence compression of spoken utterances in order to remove redundant or unnecessary words while trying to preserve the information in the original sentence. Sentence compression has been studied from formal text domain to speech domain. In text domain, (Knight and Marcu, 2000) applies noisy-channel model and decision tree approaches on this problem. (Galley and Mckeown, 2007) proposes to use a synchronous context-free grammars (SCFG) based method to compress the sentence. (Cohn and Lapata, 2008) expands the operation set by including insertion, substitution and reordering, and incorporates grammar rules. In speech domain, (Clarke and Lapata, 2008) investigates sentence compression in broadcast news using an integer linear programming approach. There is only a few existing work in spontaneous speech domains. (Liu and Liu, 2010) modeled it as a sequence labeling problem using conditional random fields model. (Liu and Liu, 2009) compared the effect of different compression methods on a meeting summarization task, but did not evaluate sentence compression itself.

We propose to use a two-step approach in this paper for sentence compression of spontaneous speech utterances. The contributions of our work are:

- Our proposed two-step approach allows us to incorporate features from local and global levels. In the first step, we adopt a similar sequence labeling method as used in (Liu and Liu, 2010), but expanded the feature set, which

[1]For speech domains, "sentences" are not clearly defined. We use sentences and utterances interchangeably when there is no ambiguity.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 166–170,
Jeju, Republic of Korea, 8-14 July 2012.

results in better performance. In the second step, we use discriminative reranking to incorporate global information about the compressed sentence candidates, which cannot be accomplished by word level labeling.

- We evaluate our methods using different metrics including word-level accuracy and F1-measure by comparing to one reference compression, and BLEU scores comparing with multiple references. We also demonstrate that training in the reranking module can be tailed to the evaluation metrics to optimize system performance.

2 Corpus

We use the same corpus as (Liu and Liu, 2010) where they annotated 2,860 summary sentences in 26 meetings from the ICSI meeting corpus (Murray et al., 2005). In their annotation procedure, filled pauses such as "uh/um" and incomplete words are removed before annotation. In the first step, 8 annotators were asked to select words to be removed to compress the sentences. In the second step, 6 annotators (different from the first step) were asked to pick the best one from the 8 compressions from the previous step. Therefore for each sentence, we have 8 human compressions, as well a best one selected by the majority of the 6 annotators in the second step. The compression ratio of the best human reference is 63.64%.

In the first step of our sentence compression approach (described below), for model training we need the reference labels for each word, which represents whether it is preserved or deleted in the compressed sentence. In (Liu and Liu, 2010), they used the labels from the annotators directly. In this work, we use a different way. For each sentence, we still use the best compression as the gold standard, but we realign the pair of the source sentence and the compressed sentence, instead of using the labels provided by annotators. This is because when there are repeated words, annotators sometimes randomly pick removed ones. However, we want to keep the patterns consistent for model training – we always label the last appearance of the repeated words as 'preserved', and the earlier ones as 'deleted'. Another difference in our processing of the corpus from the previous work is that when aligning the original and the compressed sentence, we keep filled pauses and incomplete words since they tend to appear together with disfluencies and thus provide useful information for compression.

3 Sentence Compression Approach

Our compression approach has two steps: in the first step, we use Conditional Random Fields (CRFs) to model this problem as a sequence labeling task, where the label indicates whether the word should be removed or not. We select n-best candidates ($n = 25$ in our work) from this step. In the second step we use discriminative training based on a maximum Entropy model to rerank the candidate compressions, in order to select the best one based on the quality of the whole candidate sentence, which cannot be performed in the first step.

3.1 Generate N-best Candidates

In the first step, we cast sentence compression as a sequence labeling problem. Considering that in many cases phrases instead of single words are deleted, we adopt the 'BIO' labeling scheme, similar to the name entity recognition task: "B" indicates the first word of the removed fragment, "I" represents inside the removed fragment (except the first word), and "O" means outside the removed fragment, i.e., words remaining in the compressed sentence. Each sentence with n words can be viewed as a word sequence $X_1, X_2, ..., X_n$, and our task is to find the best label sequence $Y_1, Y_2, ..., Y_n$ where Y_i is one of the three labels. Similar to (Liu and Liu, 2010), for sequence labeling we use linear-chain first-order CRFs. These models define the conditional probability of each labeling sequence given the word sequence as:

$$p(Y|X) \propto$$
$$exp \sum_{k=1}^{n} (\sum_j \lambda_j f_j(y_k, y_{k-1}, X) + \sum_i \mu_i g_i(x_k, y_k, X))$$

where f_j are transition feature functions (here first-order Markov independence assumption is used); g_i are observation feature functions; λ_j and μ_i are their corresponding weights. To train the model for this step, we use the best reference compression to obtain the reference labels (as described in Section 2).

In the CRF compression model, each word is represented by a feature vector. We incorporate most of the features used in (Liu and Liu, 2010), including unigram, position, length of utterance, part-of-speech tag as well as syntactic parse tree tags. We did not use the discourse parsing tree based features because we found they are not useful in our experiments. In this work, we further expand the feature set in order to represent the characteristics of disfluencies in spontaneous speech as well as model the adjacent output labels. The additional features we

introduced are:

- the distance to the next same word and the next same POS tag.
- a binary feature to indicate if there is a filled pause or incomplete word in the following 4-word window. We add this feature since filled pauses or incomplete words often appear after disfluent words.
- the combination of word/POS tag and its position in the sentence.
- language model probabilities: the bigram probability of the current word given the previous one, and followed by the next word, and their product. These probabilities are obtained from the Google Web 1T 5-gram.
- transition features: a combination of the current output label and the previous one, together with some observation features such as the unigram and bigrams of word or POS tag.

3.2 Discriminative Reranking

Although CRFs is able to model the dependency of adjacent labels, it does not measure the quality of the whole sentence. In this work, we propose to use discriminative training to rerank the candidates generated in the first step. Reranking has been used in many tasks to find better global solutions, such as machine translation (Wang et al., 2007), parsing (Charniak and Johnson, 2005), and disfluency detection (Zwarts and Johnson, 2011). We use a maximum Entropy reranker to learn distributions over a set of candidates such that the probability of the best compression is maximized. The conditional probability of output y given observation x in the maximum entropy model is defined as:

$$p(y|x) = \frac{1}{Z(x)} exp\left[\sum_{i=1}^{k} \lambda_i f(x,y)\right]$$

where $f(x,y)$ are feature functions and λ_i are their weighting parameters; $Z(x)$ is the normalization factor.

In this reranking model, every compression candidate is represented by the following features:

- All the bigrams and trigrams of words and POS tags in the candidate sentence.
- Bigrams and trigrams of words and POS tags in the original sentence in combination with their binary labels in the candidate sentence (delete the word or not). For example, if the original sentence is "so I should go", and the candidate compression sentence is "I should go", then "so_I_10", "so_I_should_100" are included in the features (1 means the word is deleted).
- The log likelihood of the candidate sentence based on the language model.
- The absolute difference of the compression ratio of the candidate sentence with that of the first ranked candidate. This is because we try to avoid a very large or small compression ratio, and the first candidate is generally a good candidate with reasonable length.
- The probability of the label sequence of the candidate sentence given by the first step CRFs.
- The rank of the candidate sentence in 25 best list.

For discriminative training using the n-best candidates, we need to identify the best candidate from the n-best list, which can be either the reference compression (if it exists on the list), or the most similar candidate to the reference. Since we have 8 human compressions and also want to evaluate system performance using all of them (see experiments later), we try to use multiple references in this reranking step. In order to use the same training objective (maximize the score for the single best among all the instances), for the 25-best list, if m reference compressions exist, we split the list into m groups, each of which is a new sample containing one reference as positive and several negative candidates. If no reference compression appears in 25-best list, we just keep the entire list and label the instance that is most similar to the best reference compression as positive.

4 Experiments

We perform a cross-validation evaluation where one meeting is used for testing and the rest of them are used as the training set. When evaluating the system performance, we do not consider filled pauses and incomplete words since they can be easily identified and removed. We use two different performance metrics in this study.

- Word-level accuracy and F1 score based on the minor class (removed words). This was used in (Liu and Liu, 2010). These measures are obtained by comparing with the best compression. In evaluation we map the result using 'BIO' labels from the first-step compression to binary labels that indicate a word is removed or not.

- BLEU score. BLEU is a widely used metric in evaluating machine translation systems that often use multiple references. Since there is a great variation in human compression results, and we have 8 reference compressions, we explore using BLEU for our sentence compression task. BLEU is calculated based on the precision of n-grams. In our experiments we use up to 4-grams.

Table 1 shows the averaged scores of the cross validation evaluation using the above metrics for several methods. Also shown in the table is the compression ratio of the system output. For "reference", we randomly choose one compression from 8 references, and use the rest of them as references in calculating the BLEU score. This represents human performance. The row "basic features" shows the result of using all features in (Liu and Liu, 2010) except discourse parsing tree based features, and using binary labels (removed or not). The next row uses this same basic feature set and "BIO" labels. Row "expanded features" shows the result of our expanded feature set using "BIO" label set from the first step of compression. The last two rows show the results after reranking, trained using one best reference or 8 reference compressions, respectively.

	accuracy	F1	BLEU	ratio (%)
reference	81.96	69.73	95.36	76.78
basic features (Liu and Liu, 2010)	76.44	62.11	91.08	73.49
basic features, BIO	77.10	63.34	91.41	73.22
expanded features	**79.28**	67.37	92.70	72.17
reranking train w/ 1 ref	79.01	**67.74**	91.90	70.60
reranking train w/ 8 refs	78.78	63.76	**94.21**	77.15

Table 1: Compression results using different systems.

Our result using the basic feature set is similar to that in (Liu and Liu, 2010) (their accuracy is 76.27% when compression ratio is 0.7), though the experimental setups are different: they used 6 meetings as the test set while we performed cross validation. Using the "BIO" label set instead of binary labels has marginal improvement for the three scores. From the table, we can see that our expanded feature set is able to significantly improve the result, suggesting the effectiveness of the new introduced features.

Regarding the two training settings in reranking, we find that there is no gain from reranking when using only one best compression, however, training with multiple references improves BLEU scores. This indicates the discriminative training used in maximum entropy reranking is consistent with the performance metrics. Another reason for the performance gain for this condition is that there is less data imbalance in model training (since we split the n-best list, each containing fewer negative examples). We also notice that the compression ratio after reranking is more similar to the reference. As suggested in (Napoles et al., 2011), it is not appropriate to compare compression systems with different compression ratios, especially when considering grammars and meanings. Therefore for the compression system without reranking, we generated results with the same compression ratio (77.15%), and found that using reranking still outperforms this result, 1.19% higher in BLEU score.

For an analysis, we check how often our system output contains reference compressions based on the 8 references. We found that 50.8% of system generated compressions appear in the 8 references when using CRF output with a compression ration of 77.15%; and after reranking this number increases to 54.8%. This is still far from the oracle result – for 84.7% of sentences, the 25-best list contains one or more reference sentences, that is, there is still much room for improvement in the reranking process. The results above also show that the token level measures by comparing to one best reference do not always correlate well with BLEU scores obtained by comparing with multiple references, which shows the need of considering multiple metrics.

5 Conclusion

This paper presents a 2-step approach for sentence compression: we first generate an n-best list for each source sentence using a sequence labeling method, then rerank the n-best candidates to select the best one based on the quality of the whole candidate sentence using discriminative training. We evaluate the system performance using different metrics. Our results show that our expanded feature set improves the performance across multiple metrics, and reranking is able to improve the BLEU score. In future work, we will incorporate more syntactic information in the model to better evaluate sentence quality. We also plan to perform a human evaluation for the compressed sentences, and use sentence compression in summarization.

6 Acknowledgment

This work is partly 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

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pages 173–180, Stroudsburg, PA, USA. Proceedings of ACL.

James Clarke and Mirella Lapata. 2008. *Global inference for sentence compression an integer linear programming approach*. *Journal of Artificial Intelligence Research*, 31:399–429, March.

Trevor Cohn and Mirella Lapata. 2008. *Sentence compression beyond word deletion*. In *Proceedings of COLING*.

Michel Galley and Kathleen R. Mckeown. 2007. *Lexicalized Markov grammars for sentence compression*. In *Proceedings of HLT-NAACL*.

Kevin Knight and Daniel Marcu. 2000. *Statistics-based summarization-step one: Sentence compression*. In *Proceedings of AAAI*.

Fei Liu and Yang Liu. 2009. *From extractive to abstractive meeting summaries: can it be done by sentence compression?* In *Proceedings of the ACL-IJCNLP*.

Fei Liu and Yang Liu. 2010. *Using spoken utterance compression for meeting summarization: a pilot study*. In *Proceedings of SLT*.

Gabriel Murray, Steve Renals, and Jean Carletta. 2005. *Extractive summarization of meeting recordings*. In *Proceedings of EUROSPEECH*.

Courtney Napoles, Benjamin Van Durme, and Chris Callison-Burch. 2011. *Evaluating Sentence Compression: Pitfalls and Suggested Remedies*. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 91–97, Portland, Oregon, June. Association for Computational Linguistics.

Wen Wang, A. Stolcke, and Jing Zheng. 2007. *Reranking machine translation hypotheses with structured and web-based language models*. In *Proceedings of IEEE Workshop on Speech Recognition and Understanding*, pages 159–164, Kyoto.

Simon Zwarts and Mark Johnson. 2011. *The impact of language models and loss functions on repair disfluency detection*. In *Proceedings of ACL*.

Syntactic Stylometry for Deception Detection

Song Feng **Ritwik Banerjee** **Yejin Choi**
Department of Computer Science
Stony Brook University
Stony Brook, NY 11794-4400
songfeng, rbanerjee, ychoi@cs.stonybrook.edu

Abstract

Most previous studies in computerized deception detection have relied only on shallow lexico-syntactic patterns. This paper investigates syntactic stylometry for deception detection, adding a somewhat unconventional angle to prior literature. Over four different datasets spanning from the product review to the essay domain, we demonstrate that features driven from Context Free Grammar (CFG) parse trees consistently improve the detection performance over several baselines that are based only on shallow lexico-syntactic features. Our results improve the best published result on the hotel review data (Ott et al., 2011) reaching 91.2% accuracy with 14% error reduction.

1 Introduction

Previous studies in computerized deception detection have relied only on shallow lexico-syntactic cues. Most are based on dictionary-based word counting using LIWC (Pennebaker et al., 2007) (e.g., Hancock et al. (2007), Vrij et al. (2007)), while some recent ones explored the use of machine learning techniques using simple lexico-syntactic patterns, such as n-grams and part-of-speech (POS) tags (Mihalcea and Strapparava (2009), Ott et al. (2011)). These previous studies unveil interesting correlations between certain lexical items or categories with deception that may not be readily apparent to human judges. For instance, the work of Ott et al. (2011) in the hotel review domain results in very insightful observations that deceptive reviewers tend to use verbs and personal pronouns (e.g., "I", "my") more often, while truthful reviewers tend to use more of nouns, adjectives, prepositions. In parallel to these shallow lexical patterns, might there be deep syntactic structures that are lurking in deceptive writing?

This paper investigates syntactic stylometry for deception detection, adding a somewhat unconventional angle to prior literature. Over four different datasets spanning from the product review domain to the essay domain, we find that features driven from Context Free Grammar (CFG) parse trees consistently improve the detection performance over several baselines that are based only on shallow lexico-syntactic features. Our results improve the best published result on the hotel review data of Ott et al. (2011) reaching 91.2% accuracy with 14% error reduction. We also achieve substantial improvement over the essay data of Mihalcea and Strapparava (2009), obtaining upto 85.0% accuracy.

2 Four Datasets

To explore different types of deceptive writing, we consider the following four datasets spanning from the product review to the essay domain:

I. TripAdvisor—Gold: Introduced in Ott et al. (2011), this dataset contains 400 truthful reviews obtained from `www.tripadviser.com` and 400 deceptive reviews gathered using Amazon Mechanical Turk, evenly distributed across 20 Chicago hotels.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 171–175,
Jeju, Republic of Korea, 8-14 July 2012.

TripAdvisor–Gold		TripAdvisor–Heuristic	
Deceptive	Truthful	Deceptive	Truthful
NP^PP → DT NNP NNP NNP	S^ROOT → VP .	NP^S → PRP	VP^S → VBZ NP
SBAR^NP → S	NP^NP → $ CD	SBAR^S → WHADVP S	NP^NP → NNS
NP^VP → NP SBAR	PRN^NP → LRB NP RRB	VP^S → VBD PP	WHNP^SBAR → WDT
NP^NP → PRP$ NN	NP^NP → NNS	S^SBAR → NP VP	NP^NP → NP PP PP
NP^S → DT NNP NNP NNP	NP^S → NN	S^ROOT → PP NP VP .	NP^S → EX
VP^S → VBG PP	NP^PP → DT NNP	VP^S → VBD S	NX^NX → JJ NN
NP^PP → PRP$ NN	NP^PP → CD NNS	NP^S → NP CC NP	NP^NP → NP PP
VP^S → MD ADVP VP	NP^NP → NP PRN	NP^S → PRP$ NN	VP^S → VBZ RB NP
VP^S → TO VP	PRN^NP → LRB PP RRB	NP^PP → DT NNP	PP^NP → IN NP
ADJP^NP → RBS JJ	NP^NP → CD NNS	NP^PP → PRP$ NN	PP^ADJP → TO NP

Table 1: Most discriminative rewrite rules ($\hat{r}$): hotel review datasets

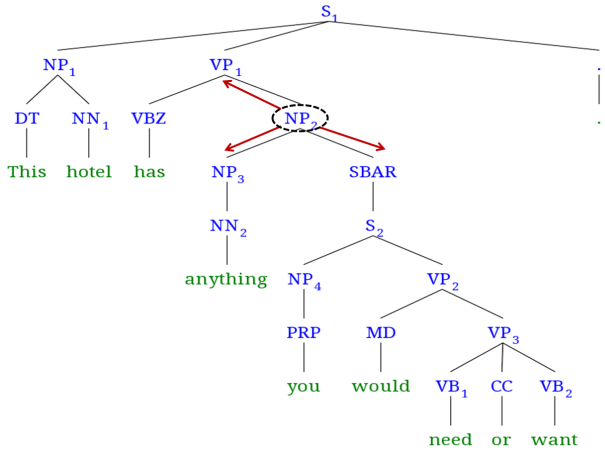


Figure 1: Parsed trees

II. TripAdvisor—Heuristic: This dataset contains 400 truthful and 400 deceptive reviews harvested from `www.tripadviser.com`, based on fake review detection heuristics introduced in Feng et al. (2012).[1]

III. Yelp: This dataset is our own creation using `www.yelp.com`. We collect 400 *filtered* reviews and 400 *displayed* reviews for 35 Italian restaurants with average ratings in the range of [3.5, 4.0]. Class labels are based on the meta data, which tells us whether each review is *filtered* by Yelp's automated review filtering system or not. We expect that *filtered* reviews roughly correspond to deceptive reviews, and *displayed* reviews to truthful ones, but not without considerable noise. We only collect 5-star reviews to avoid unwanted noise from varying degree of sentiment.

IV. Essays: Introduced in Mihalcea and Strapparava (2009), this corpus contains truthful and deceptive essays collected using Amazon Mechanic Turk for the following three topics: "Abortion" (100 essays per class), "Best Friend" (98 essays per class), and "Death Penalty" (98 essays per class).

3 Feature Encoding

Words Previous work has shown that bag-of-words are effective in detecting domain-specific deception (Ott et al., 2011; Mihalcea and Strapparava, 2009). We consider unigram, bigram, and the union of the two as features.

Shallow Syntax As has been used in many previous studies in stylometry (e.g., Argamon-Engelson et al. (1998), Zhao and Zobel (2007)), we utilize part-of-speech (POS) tags to encode shallow syntactic information. Note that Ott et al. (2011) found that even though POS tags are effective in detecting fake product reviews, they are not as effective as words. Therefore, we strengthen POS features with unigram features.

Deep syntax We experiment with four different encodings of production rules based on the Probabilistic Context Free Grammar (PCFG) parse trees as follows:

- r: unlexicalized production rules (i.e., all production rules except for those with terminal nodes), e.g., $NP_2 \rightarrow NP_3$ SBAR.
- $r*$: lexicalized production rules (i.e., *all* production rules), e.g., PRP → "you".
- $\hat{r}$: unlexicalized production rules combined with the grandparent node, e.g., NP_2^VP

[1] Specifically, using the notation of Feng et al. (2012), we use data created by Strategy-$dist\Phi$ heuristic, with $H_S, \mathcal{S}$ as deceptive and $H'_S, \mathcal{T}$ as truthful.

		TripAdvisor		Yelp	Essay		
		Gold	Heur		Abort	BstFr	Death
words	unigram	*88.4*	**74.4**	59.9	*70.0*	*77.0*	**67.4**
	bigram	85.8	71.5	**60.7**	71.5	79.5	55.5
	uni + bigram	**89.6**	73.8	60.1	**72.0**	**81.5**	65.5
shallow syntax +words	pos(n=1) + unigram	87.4	74.0	62.0	70.0	80.0	66.5
	pos(n=2) + unigram	88.6	74.6	59.0	67.0	82.0	66.5
	pos(n=3) + unigram	88.6	74.6	59.3	67.0	82.0	66.5
deep syntax	r	78.5	65.3	56.9	62	67.5	55.5
	$\hat{r}$	74.8	65.3	56.5	58.5	65.5	56.0
	$r*$	89.4	74.0	64.0	70.1	77.5	66.0
	$\hat{r}*$	90.4	75	63.5	71.0	78	67.5
deep syntax +words	r + unigram	89.0	74.3	62.3	76.5	82.0	69.0
	$\hat{r}$ + unigram	88.5	74.3	62.5	**77.0**	81.5	70.5
	$r*$ + unigram	90.3	75.4	**64.3**	74.0	**85.0**	**71.5**
	$\hat{r}*$ + unigram	**91.2**	**76.6**	62.1	76.0	84.5	71.0

Table 2: Deception Detection Accuracy (%).

$_1 \rightarrow \text{NP}_3$ SBAR.

- $\hat{r}*$: lexicalized production rules (i.e., *all* production rules) combined with the grandparent node, e.g., PRP$\hat{}$NP $_4 \rightarrow$ "you".

4 Experimental Results

For all classification tasks, we use SVM classifier, 80% of data for training and 20% for testing, with 5-fold cross validation.[2] All features are encoded as tf-idf values. We use Berkeley PCFG parser (Petrov and Klein, 2007) to parse sentences. Table 2 presents the classification performance using various features across four different datasets introduced earlier.[3]

4.1 TripAdvisor–Gold

We first discuss the results for the TripAdvisor–Gold dataset shown in Table 2. As reported in Ott et al. (2011), bag-of-words features achieve surprisingly high performance, reaching upto 89.6% accuracy. Deep syntactic features, encoded as $\hat{r}*$ slightly improves this performance, achieving 90.4% accuracy. When these syntactic features are combined with unigram features, we attain the best performance of 91.2% accuracy, yielding 14% error reduction over the word-only features.

Given the power of word-based features, one might wonder, whether the PCFG driven features are being useful only due to their lexical production rules. To address such doubts, we include experiments with unlexicalized rules, r and $\hat{r}$. These features achieve 78.5% and 74.8% accuracy respectively, which are significantly higher than that of a random baseline (∼50.0%), confirming statistical differences in deep syntactic structures. See Section 4.4 for concrete exemplary rules.

Another question one might have is whether the performance gain of PCFG features are mostly from local sequences of POS tags, indirectly encoded in the production rules. Comparing the performance of [shallow syntax+words] and [deep syntax+words] in Table 2, we find statistical evidence that deep syntax based features offer information that are not available in simple POS sequences.

4.2 TripAdvisor–Heuristic & Yelp

The performance is generally lower than that of the previous dataset, due to the noisy nature of these datasets. Nevertheless, we find similar trends as those seen in the TripAdvisor–Gold dataset, with respect to the relative performance differences across different approaches. The sig-

[2]We use LIBLINEAR (Fan et al., 2008) with L2-regulization, parameter optimized over the 80% training data (3 folds for training, 1 fold for testing).

[3]Numbers in *italic* are classification results reported in Ott et al. (2011) and Mihalcea and Strapparava (2009).

TripAdvisor–Gold		TripAdvisor–Heur	
Decep	Truth	Decep	Truth
VP	PRN	VP	PRN
SBAR	QP	WHADVP	NX
WHADVP	S	SBAR	WHNP
ADVP	PRT	WHADJP	ADJP
CONJP	UCP	INTJ	WHPP

Table 3: Most discriminative phrasal tags in PCFG parse trees: TripAdvisor data.

nificance of these results comes from the fact that these two datasets consists of real (fake) reviews in the wild, rather than manufactured ones that might invite unwanted signals that can unexpectedly help with classification accuracy. In sum, these results indicate the existence of the statistical signals hidden in deep syntax even in real product reviews with noisy gold standards.

4.3 Essay

Finally in Table 2, the last dataset Essay confirms the similar trends again, that the deep syntactic features consistently improve the performance over several baselines based only on shallow lexico-syntactic features. The final results, reaching accuracy as high as 85%, substantially outperform what has been previously reported in Mihalcea and Strapparava (2009). How robust are the syntactic cues in the cross topic setting? Table 4 compares the results of Mihalcea and Strapparava (2009) and ours, demonstrating that syntactic features achieve substantially and surprisingly more robust results.

4.4 Discriminative Production Rules

To give more concrete insights, we provide 10 most discriminative unlexicalized production rules (augmented with the grand parent node) for each class in Table 1. We order the rules based on the feature weights assigned by LIBLINEAR classifier. Notice that the two production rules in bolds — [SBAR^NP → S] and [NP ^VP → NP SBAR] — are parts of the parse tree shown in Figure 1, whose sentence is taken from an actual fake review. Table 3 shows the most discriminative phrasal tags in the PCFG parse trees for each class. Interestingly, we find more frequent use of VP, SBAR (clause introduced by subordinating conjunction), and WHADVP in deceptive reviews than truthful reviews.

training:	**A** & **B**	**A** & **D**	**B** & **D**
testing:	**D**eathPen	**B**estFrn	**A**bortion
M&S 2009	58.7	58.7	62.0
$r*$	66.8	70.9	69.0

Table 4: Cross topic deception detection accuracy: Essay data

5 Related Work

Much of the previous work for detecting deceptive product reviews focused on related, but slightly different problems, e.g., detecting duplicate reviews or review spams (e.g., Jindal and Liu (2008), Lim et al. (2010), Mukherjee et al. (2011), Jindal et al. (2010)) due to notable difficulty in obtaining gold standard labels.[4] The Yelp data we explored in this work shares a similar spirit in that gold standard labels are harvested from existing meta data, which are not guaranteed to align well with true hidden labels as to deceptive v.s. truthful reviews. Two previous work obtained more precise gold standard labels by hiring Amazon turkers to write deceptive articles (e.g., Mihalcea and Strapparava (2009), Ott et al. (2011)), both of which have been examined in this study with respect to their syntactic characteristics. Although we are not aware of any prior work that dealt with syntactic cues in deceptive writing directly, prior work on hedge detection (e.g., Greene and Resnik (2009), Li et al. (2010)) relates to our findings.

6 Conclusion

We investigated syntactic stylometry for deception detection, adding a somewhat unconventional angle to previous studies. Experimental results consistently find statistical evidence of deep syntactic patterns that are helpful in discriminating deceptive writing.

[4]It is not possible for a human judge to tell with full confidence whether a given review is a fake or not.

References

S. Argamon-Engelson, M. Koppel, and G. Avneri. 1998. Style-based text categorization: What newspaper am i reading. In *Proc. of the AAAI Workshop on Text Categorization*, pages 1–4.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

S. Feng, L. Xing, Gogar A., and Y. Choi. 2012. Distributional footprints of deceptive product reviews. In *Proceedings of the 2012 International AAAI Conference on WebBlogs and Social Media*, June.

S. Greene and P. Resnik. 2009. More than words: Syntactic packaging and implicit sentiment. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 503–511. Association for Computational Linguistics.

J.T. Hancock, L.E. Curry, S. Goorha, and M. Woodworth. 2007. On lying and being lied to: A linguistic analysis of deception in computer-mediated communication. *Discourse Processes*, 45(1):1–23.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 219–230, New York, NY, USA. ACM.

Nitin Jindal, Bing Liu, and Ee-Peng Lim. 2010. Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management*, pages 1549–1552.

X. Li, J. Shen, X. Gao, and X. Wang. 2010. Exploiting rich features for detecting hedges and their scope. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning—Shared Task*, pages 78–83. Association for Computational Linguistics.

Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, and Hady Wirawan Lauw. 2010. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 939–948, New York, NY, USA. ACM.

R. Mihalcea and C. Strapparava. 2009. The lie detector: Explorations in the automatic recognition of deceptive language. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 309–312. Association for Computational Linguistics.

Arjun Mukherjee, Bing Liu, Junhui Wang, Natalie S. Glance, and Nitin Jindal. 2011. Detecting group review spam. In *Proceedings of the 20th International Conference on World Wide Web (Companion Volume)*, pages 93–94.

Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 309–319, Portland, Oregon, USA, June. Association for Computational Linguistics.

J.W. Pennebaker, C.K. Chung, M. Ireland, A. Gonzales, and R.J. Booth. 2007. The development and psychometric properties of liwc2007. *Austin, TX, LIWC. Net.*

S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of NAACL HLT 2007*, pages 404–411.

A. Vrij, S. Mann, S. Kristen, and R.P. Fisher. 2007. Cues to deception and ability to detect lies as a function of police interview styles. *Law and human behavior*, 31(5):499–518.

Ying Zhao and Justin Zobel. 2007. Searching with style: authorship attribution in classic literature. In *Proceedings of the thirtieth Australasian conference on Computer science - Volume 62*, ACSC '07, pages 59–68, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Transforming Standard Arabic to Colloquial Arabic

Emad Mohamed, Behrang Mohit and Kemal Oflazer

Carnegie Mellon University - Qatar
Doha, Qatar
emohamed@qatar.cmu.edu, behrang@cmu.edu, ko@cs.cmu.edu

Abstract

We present a method for generating Colloquial Egyptian Arabic (CEA) from morphologically disambiguated Modern Standard Arabic (MSA). When used in POS tagging, this process improves the accuracy from 73.24% to 86.84% on unseen CEA text, and reduces the percentage of out-of-vocabulary words from 28.98% to 16.66%. The process holds promise for any NLP task targeting the dialectal varieties of Arabic; e.g., this approach may provide a cheap way to leverage MSA data and morphological resources to create resources for colloquial Arabic to English machine translation. It can also considerably speed up the annotation of Arabic dialects.

1. Introduction

Most of the research on Arabic is focused on Modern Standard Arabic. Dialectal varieties have not received much attention due to the lack of dialectal tools and annotated texts (Duh and Kirchoff, 2005). In this paper, we present a rule-based method to generate Colloquial Egyptian Arabic (CEA) from Modern Standard Arabic (MSA), relying on segment-based part-of-speech tags. The transformation process relies on the observation that dialectal varieties of Arabic differ mainly in the use of affixes and function words while the word stem mostly remains unchanged. For example**,** given the Buckwalter-encoded MSA sentence "*AlAxwAn Almslm**wn lm yfwzwA fy** AlAntxbAt*" the rules produce "*AlAxwAn Almslm**yn mfAzw$ f** AlAntxAbAt*" (الاخوان المسلمين مفازوش ف الانتخابات, The Muslim Brotherhood did not win the elections). The availability of segment-based part-of-speech tags is essential since many of the affixes in MSA are ambiguous. For example, ***lm*** could be either a negative particle or a question work, and the word ***AlAxwAn*** could be either made of two segments (***Al+<xwAn***, the brothers), or three segments (***Al+>xw+An***, the two brothers).

We first introduce the transformation rules, and show that in many cases it is feasible to transform MSA to CEA, although there are cases that require much more than POS tags. We then provide a typical case in which we utilize the transformed text of the Arabic Treebank (Bies and Maamouri, 2003) to build a part-of-speech tagger for CEA. The tagger improves the accuracy of POS tagging on authentic Egyptian Arabic by 13% absolute (from 73.24% to 86.84%) and reduces the percentage of out-of-vocabulary words from 28.98% to 16.66%.

2. MSA to CEA Conversion Rules

Table 1 shows a sentence in MSA and its CEA counterpart. Both can be translated into: *"We did not write it for them.*" MSA has three words while CEA is more synthetic as the preposition and the negative particle turn into clitics. Table 1 illustrates the end product of one of the Imperfect transformation rules, namely the case where the Imperfect Verb is preceded by the negative particle *lm*.

	Arabic	**Buckwalter**
MSA	لم نكتبها لهن	lm nktbhA lhn
CEA	مكتبنهلهمش	mktbnhlhm$
English	We did not write it for them	

Table 1: a sentence in MSA and CEA

Our 103 rules cover nominals (number and case affixes), verbs (tense, number, gender, and modality), pronouns (number and gender), and demonstrative pronouns (number and gender).

The rules also cover certain lexical items as 400 words in MSA have been converted to their com-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 176–180,
Jeju, Republic of Korea, 8-14 July 2012.

mon CEA counterparts. Examples of lexical conversions include ***ZlAm*** and ***Dlmp*** (darkness), ***rjl*** and ***rAjl*** (man), ***rjAl*** and ***rjAlp*** (men), and ***kvyr*** and ***ktyr*** (many), where the first word is the MSA version and the second is the CEA version.

Many of the lexical mappings are ambiguous. For example, the word ***rjl*** can either mean ***man*** or ***leg***. When it means *man*, the CEA form is ***rAjl***, but the word for *leg* is the same in both MSA and CEA. While they have different vowel patterns (***rajul*** and ***rijol*** respectively), the vowel information is harder to get correctly than POS tags. The problem may arise especially when dealing with raw data for which we need to provide POS tags (and vowels) so we may be able to convert it to the colloquial form. Below, we provide two sample rules:

The imperfect verb is used, inter alia, to express the negated past, for which CEA uses the perfect verb. What makes things more complicated is that CEA treats negative particles and prepositional phrases as clitics. An example of this is the word ***mktbthlhm$*** (I did not write it for them) in Table 1 above. It is made of the negative particle ***m***, the stem ***ktb*** (to write), the object pronoun ***h,*** the preposition ***l***, the pronoun ***hm*** (them) and the negative particle ***$.*** Figure 1, and the following steps show the conversions of ***lm nktbhA lhm*** to ***mktbnhAlhm$:***

1. Replace the negative word ***lm*** with one of the prefixes ***m***, ***mA*** or the word ***mA.***
2. Replace the Imperfect Verb prefix with its Perfect Verb suffix counterpart. For example, the IV first person singular subject prefix > turns into ***t*** in the PV.
3. If the verb is followed by a prepositional phrase headed by the preposition ***l*** that contains a pronominal object, convert the preposition to a prepositional clitic.
4. Transform the dual to plural and the plural feminine to plural masculine.
5. Add the negative suffix ***$*** (or the variant ***$y***, which is less probable)

As alluded to in 1) above, given that colloquial orthography is not standardized, many affixes and clitics can be written in different ways. For example, the word ***mktbnhlhm$,*** can be written in 24 ways. All these forms are legal and possible, as attested by their existence in a CEA corpus (the Arabic Online Commentary Dataset v1.1), which we also use for building a language model later.

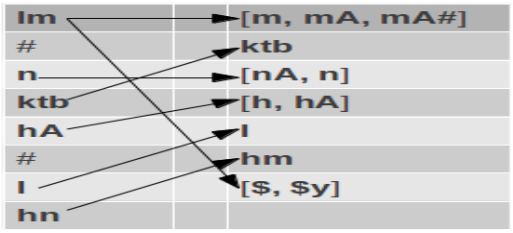


Figure 1: One negated IV form in MSA can generate 24 (3x2x2x2) possible forms in CEA

MSA possessive pronouns inflect for gender, number (singular, dual, and plural), and person. In CEA, there is no distinction between the dual and the plural, and a single pronoun is used for the plural feminine and masculine. The three MSA forms *ktAb**hm***, *ktAb**hmA*** and *ktAb**hn*** (***their*** *book* for the masculine plural, the dual, and the feminine plural respectively) all collapse to *ktAb**hm***.

Table 2 has examples of some other rules we have applied. We note that the stem, in bold, hardly changes, and that the changes mainly affect function segments. The last example is a lexical rule in which the stem has to change.

Rule	**MSA**	**CEA**
Future	swf y**ktb**	Hy**ktb**/hy**ktb**
Future_NEG	ln >**ktb**	m$ h**ktb**/ m$ H**ktb**
IV	y**ktb**wn	by**ktb**w/ b**ktb**w/ b**ktb**wA
Passive	**ktb**	An**ktb**/ At**ktb**
NEG_PREP	lys **mn**hn	m**mn**hm$
Lexical	**trk**hmA	**sAb**hm

Table 2: Examples of Conversion Rules.

3. POS Tagging Egyptian Arabic

We use the conversion above to build a POS tagger for Egyptian Arabic. We follow Mohamed and Kuebler (2010) in using whole word tagging, i.e., without any word segmentation. We use the Columbia Arabic Treebank 6-tag tag set: PRT (Particle), NOM (Nouns, Adjectives, and Adverbs), PROP (Proper Nouns), VRB (Verb), VRB-pass (Passive Verb), and PNX (Punctuation) (Habash and Roth, 2009). For example, the word ***wHnktblhm*** (*and we will write to them,* وحنكتبلهم) receives the tag ***PRT+PRT+VRB+PRT+NOM.*** This results in 58 composite tags, 9 of which occur 5 times or less in the converted ECA training set.

We converted two sections of the Arabic Treebank (ATB): p2v3 and p3v2. For all the POS tagging experiments, we use the memory-based POS tagger (MBT) (Daelemans *et al.*, 1996) The best results, tuned on a dev set, were obtained, in non-exhaustive search, with the Modified Value Difference Metric as a distance metric and with *k* (the number of nearest neighbors) = 25. For known words, we use the IGTree algorithm and 2 words to the left, their POS tags, the focus word and its list of possible tags, 1 right context word and its list of possible tags as features. For unknown words, we use the IB1 algorithm and the word itself, its first 5 and last 3 characters, 1 left context word and its POS tag, and 1 right context word and its list of possible tags as features.

3.1. Development and Test Data

As a development set, we use 100 user-contributed comments (2757 words) from the website ***masrawy.com***, which were judged to be highly colloquial. The test set contains 192 comments (7092 words) from the same website with the same criterion. The development and test sets were hand-annotated with composite tags as illustrated above by two native Arabic-speaking students.

The test and development sets contained spelling errors (mostly run-on words). The most common of these is the vocative particle ***yA***, which is usually attached to following word (e.g. ***yArAjl***, (you man, ياراجل)). It is not clear whether it should be treated as a proclitic, since it also occurs as a separate word, which is the standard way of writing. The same holds true for the variation between the letters ***** and ***z***, (ذ and ز in Arabic) which are pronounced exactly the same way in CEA to the extent that the substitution may not be considered a spelling error.

3.2. Experiments and Results

We ran five experiments to test the effect of MSA to CEA conversion on POS tagging: (a) **Standard**, where we train the tagger on the ATB MSA data, (b) **3-gram LM**, where for each MSA sentence we generate all transformed sentences (see Section 2.1 and Figure 1) and pick the most probable sentence according to a trigram language model built from an 11.5 million words of user contributed comments.[1] This corpus is highly dialectal Egyptian Arabic, but like all similar collections, it is diglossic and demonstrates a high degree of code-switching between MSA and CEA. We use the SRILM toolkit (Stolcke, 2002) for language modeling and sentence scoring, (c) **Random**, where we choose a random sentence from all the correct sentences generated for each MSA sentence, (d) **Hybrid**, where we combine the data in a) with the best settings (as measured on the dev set) using the converted colloquial data (namely experiment c). Hybridization is necessary since most Arabic data in blogs and comments are a mix of MSA and CEA, and (e) **Hybrid + dev,** where we enrich the Hybrid training set with the dev data.

We use the following metrics for evaluation: **KWA**: Known Word Accuracy (%), **UWA**: Unknown Word Accuracy (%), **TA**: Total Accuracy (%), and **UW**: unknown words (%) in the respective set in the respective experiment. Table 3(a) presents the results on the development set while Table 3(b) the results on the test set.

Experiment	KWA	UWA	TA	UW
(a) Standard	92.75	39.68	75.77	31.99
(b) 3-gram LM	89.12	43.46	76.21	28.29
(c) Random	92.36	43.51	79.25	26.84
(d) Hybrid	**94.13**	**52.22**	**84.87**	**22.09**

Table 3(a): POS results on the development set.

We notice that randomly selecting a sentence from the correct generated sentences yields better results than choosing the most probable sentence according to a language model. The reason for this may be that randomization guarantees more coverage of the various forms. We have found that the vocabulary size (the number of unique word types) for the training set generated for the **Random** experiment is considerably larger than the vocabulary size for the 3-gram LM experiment (55367 unique word types in **Random** versus 51306 in **3-gram LM**), which results in a drop of 4.6% absolute in the percentage of unknown words: 27.31% versus 22.30%). This drop in the percentage of unknown words may indicate that generating all possible variations of CEA may be more useful than using a language model in general. Even in a CEA corpus of 35 million words, one third of the words generated by the rules are not in the corpus, while many

[1]Available from http://www.cs.jhu.edu/~ozaidan/AOC

of these are in both the test set and the development set.

Experiment	KWA	UWA	TA	UW
(a) Standard	89.03	40.67	73.24	28.98
(b) 3-gram LM	84.33	47.70	74.32	27.31
(c) Random	90.24	48.90	79.67	22.70
(d) Hybrid	92.22	53.92	83.81	19.45
(e) Hybrid+dev	**94.87**	**56.46**	**86.84**	**16.66**

Table 3(b): POS results on the test set

We also notice that the conversion alone improves tagging accuracy from 75.77% to 79.25% on the development set, and from 73.24% to 79.67% on the test set. Combining the original MSA and the best scoring converted data (Random) raises the accuracies to 84.87% and 83.81% respectively. The percentage of unknown words drops from 29.98% to 19.45% in the test set when we used the hybrid data. The fact that the percentage of unknown words drops further to 16.66% in the **Hybrid+dev** experiment points out the authentic colloquial data contains elements that have not been captured using conversion alone.

4. Related Work

To the best of our knowledge, ours is the first work that generates CEA automatically from morphologically disambiguated MSA, but Habash et al. (2005) discussed root and pattern morphological analysis and generation of Arabic dialects within the MAGED morphological analyzer. MAGED incorporates the morphology, phonology, and orthography of several Arabic dialects. Diab *et al*. (2010) worked on the annotation of dialectal Arabic through the COLABA project, and they used the (manually) annotated resources to facilitate the incorporation of the dialects in Arabic information retrieval.

Duh and Kirchhoff (2005) successfully designed a POS tagger for CEA that used an MSA morphological analyzer and information gleaned from the intersection of several Arabic dialects. This is different from our approach for which POS tagging is only an application. Our focus is to use any existing MSA data to generate colloquial Arabic resources that can be used in virtually any NLP task.

At a higher level, our work resembles that of Kundu and Roth (2011), in which they chose to adapt the text rather than the model. While they adapted the test set, we do so at the training set level.

5. Conclusions and Future Work

We have a presented a method to convert Modern Standard Arabic to Egyptian Colloquial Arabic with an example application to the POS tagging task. This approach may provide a cheap way to leverage MSA data and morphological resources to create resources for colloquial Arabic to English machine translation, for example.

While the rules of conversion were mainly morphological in nature, they have proved useful in handling colloquial data. However, morphology alone is not enough for handling key points of difference between CEA and MSA. While CEA is mainly an SVO language, MSA is mainly VSO, and while demonstratives are pre-nominal in MSA, they are post-nominal in CEA. These phenomena can be handled only through syntactic conversion. We expect that converting a dependency-based treebank to CEA can account for many of the phenomena part-of-speech tags alone cannot handle

We are planning to extend the rules to other linguistic phenomena and dialects, with possible applications to various NLP tasks for which MSA annotated data exist. When no gold standard segment-based POS tags are available, tools that produce segment-based annotation can be used, e.g. segment-based POS tagging (Mohamed and Kuebler, 2010) or MADA (Habash et al, 2009), although these are not expected to yield the same results as gold standard part-of-speech tags.

Acknowledgements

This publication was made possible by a NPRP grant (NPRP 09-1140-1-177) from the Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the authors.

We thank the two native speaker annotators and the anonymous reviewers for their instructive and enriching feedback.

References

Bies, Ann and Maamouri, Mohamed (2003). Penn Arabic Treebank guidelines. Technical report, LDC, University of Pennsylvania.

Buckwalter, T. (2002). Arabic Morphological Analyzer (AraMorph). Version 1.0. Linguistic Data Consortium, catalog number LDC2002L49 and ISBN 1-58563-257- 0

Daelemans, Walter and van den Bosch, Antal (2005). Memory Based Language Processing. Cambridge University Press.

Daelemans, Walter; Zavrel, Jakub; Berck, Peter, and Steven Gillis (1996). MBT: A memory-based part of speech tagger-generator. In Eva Ejerhed and Ido Dagan, editors, Proceedings of the 4th Workshop on Very Large Corpora, pages 14–27, Copenhagen, Denmark.

Diab, Mona; Habash, Nizar; Rambow, Owen; Altantawy, Mohamed, and Benajiba, Yassine. COLABA: Arabic Dialect Annotation and Processing. LREC 2010.

Duh, K. and Kirchhoff, K. (2005). POS Tagging of Dialectal Arabic: A Minimally Supervised Approach. Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages, Ann Arbor, June 2005.

Habash, Nizar; Rambow, Own and Kiraz, George (2005). Morphological analysis and generation for Arabic dialects. Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages, pages 17–24, Ann Arbor, June 2005

Habash, Nizar and Roth, Ryan. CATiB: The Columbia Arabic Treebank. Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 221–224, Singapore, 4 August 2009. c 2009 ACL and AFNLP

Habash, Nizar, Owen Rambow and Ryan Roth. MADA+TOKAN: A Toolkit for Arabic Tokenization, Diacritization, Morphological Disambiguation, POS Tagging, Stemming and Lemmatization. In Proceedings of the 2nd International Conference on Arabic Language Resources and Tools (MEDAR), Cairo, Egypt, 2009

Kundu, Gourab abd Roth, Don (2011). Adapting Text instead of the Model: An Open Domain Approach. Proceedings of the Fifteenth Conference on Computational Natural Language Learning, pages 229–237,Portland, Oregon, USA, 23–24 June 2011

Mohamed, Emad. and Kuebler, Sandra (2010). Is Arabic Part of Speech Tagging Feasible Without Word Segmentation? Proceedings of HLT-NAACL 2010, Los Angeles, CA.

Stolcke, A. (2002). SRILM - an extensible language modeling toolkit. In Proc. of ICSLP, Denver, Colorado

Corpus-based interpretation of instructions in virtual environments

Luciana Benotti[1] **Martín Villalba**[1] **Tessa Lau**[2] **Julián Cerruti**[3]
[1] FaMAF, Medina Allende s/n, Universidad Nacional de Córdoba, Córdoba, Argentina
[2]IBM Research – Almaden, 650 Harry Road, San Jose, CA 95120 USA
[3]IBM Argentina, Ing. Butty 275, C1001AFA, Buenos Aires, Argentina
{benotti,villalba}@famaf.unc.edu.ar, tessalau@us.ibm.com, jcerruti@ar.ibm.com

Abstract

Previous approaches to instruction interpretation have required either extensive domain adaptation or manually annotated corpora. This paper presents a novel approach to instruction interpretation that leverages a large amount of unannotated, easy-to-collect data from humans interacting with a virtual world. We compare several algorithms for automatically segmenting and discretizing this data into (utterance, reaction) pairs and training a classifier to predict reactions given the next utterance. Our empirical analysis shows that the best algorithm achieves 70% accuracy on this task, with no manual annotation required.

1 Introduction and motivation

Mapping instructions into automatically executable actions would enable the creation of natural language interfaces to many applications (Lau et al., 2009; Branavan et al., 2009; Orkin and Roy, 2009). In this paper, we focus on the task of navigation and manipulation of a virtual environment (Vogel and Jurafsky, 2010; Chen and Mooney, 2011).

Current symbolic approaches to the problem are brittle to the natural language variation present in instructions and require intensive rule authoring to be fit for a new task (Dzikovska et al., 2008). Current statistical approaches require extensive manual annotations of the corpora used for training (MacMahon et al., 2006; Matuszek et al., 2010; Gorniak and Roy, 2007; Rieser and Lemon, 2010). Manual annotation and rule authoring by natural language engineering experts are bottlenecks for developing conversational systems for new domains.

This paper proposes a fully automated approach to interpreting natural language instructions to complete a task in a virtual world based on unsupervised recordings of human-human interactions performing that task in that virtual world. Given unannotated corpora collected from humans following other humans' instructions, our system automatically segments the corpus into labeled training data for a classification algorithm. Our interpretation algorithm is based on the observation that similar instructions uttered in similar contexts should lead to similar actions being taken in the virtual world. Given a previously unseen instruction, our system outputs actions that can be directly executed in the virtual world, based on what humans did when given similar instructions in the past.

2 Corpora situated in virtual worlds

Our environment consists of six virtual worlds designed for the natural language generation shared task known as the GIVE Challenge (Koller et al., 2010), where a pair of partners must collaborate to solve a task in a 3D space (Figure 1). The "instruction follower" (IF) can move around in the virtual world, but has no knowledge of the task. The "instruction giver" (IG) types instructions to the IF in order to guide him to accomplish the task. Each corpus contains the IF's actions and position recorded every 200 milliseconds, as well as the IG's instructions with their timestamps.

We used two corpora for our experiments. The C_m corpus (Gargett et al., 2010) contains instructions given by multiple people, consisting of 37 games spanning 2163 instructions over 8:17 hs. The

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 181–186,
Jeju, Republic of Korea, 8-14 July 2012.

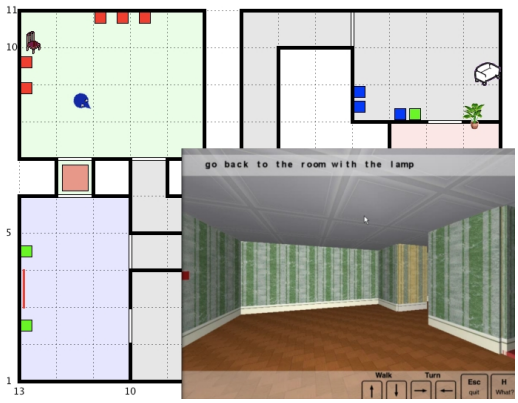


Figure 1: A screenshot of a virtual world. The world consists of interconnecting hallways, rooms and objects

C_s corpus (Benotti and Denis, 2011), gathered using a single IG, is composed of 63 games and 3417 instructions, and was recorded in a span of 6:09 hs. It took less than 15 hours to collect the corpora through the web and the subjects reported that the experiment was fun.

While the environment is restricted, people describe the same route and the same objects in extremely different ways. Below are some examples of instructions from our corpus all given for the same route shown in Figure 1.

1) out
2) walk down the passage
3) nowgo [sic] *to the pink room*
4) back to the room with the plant
5) Go through the door on the left
6) go through opening with yellow wall paper

People describe routes using landmarks (4) or specific actions (2). They may describe the same object differently (5 vs 6). Instructions also differ in their scope (3 vs 1). Thus, even ignoring spelling and grammatical errors, navigation instructions contain considerable variation which makes interpreting them a challenging problem.

3 Learning from previous interpretations

Our algorithm consists of two phases: annotation and interpretation. *Annotation* is performed only once and consists of automatically associating each IG instruction to an IF reaction. *Interpretation* is performed every time the system receives an instruction and consists of predicting an appropriate reaction given reactions observed in the corpus.

Our method is based on the assumption that a reaction captures the semantics of the instruction that caused it. Therefore, if two utterances result in the same reaction, they are paraphrases of each other, and similar utterances should generate the same reaction. This approach enables us to predict reactions for previously-unseen instructions.

3.1 Annotation phase

The key challenge in learning from massive amounts of easily-collected data is to automatically annotate an unannotated corpus. Our annotation method consists of two parts: first, *segmenting* a low-level interaction trace into utterances and corresponding reactions, and second, *discretizing* those reactions into canonical action sequences.

Segmentation enables our algorithm to learn from traces of IFs interacting directly with a virtual world. Since the IF can move freely in the virtual world, his actions are a stream of continuous behavior. Segmentation divides these traces into reactions that follow from each utterance of the IG. Consider the following example starting at the situation shown in Figure 1:

IG(1): go through the yellow opening
IF(2): [walks out of the room]
IF(3): [turns left at the intersection]
IF(4): [enters the room with the sofa]
IG(5): stop

It is not clear whether the IF is doing $\langle 3, 4 \rangle$ because he is reacting to 1 or because he is being proactive. While one could manually annotate this data to remove extraneous actions, our goal is to develop automated solutions that enable learning from massive amounts of data.

We decided to approach this problem by experimenting with two alternative formal definitions: 1) a strict definition that considers the maximum reaction according to the IF *behavior*, and 2) a loose definition based on the empirical observation that, in situated interaction, most instructions are constrained by the current *visually* perceived affordances (Gibson, 1979; Stoia et al., 2006).

We formally define *behavior segmentation* (Bhv) as follows. A reaction r_k to an instruction u_k begins

right after the instruction u_k is uttered and ends right before the next instruction u_{k+1} is uttered. In the example, instruction 1 corresponds to $\langle 2, 3, 4 \rangle$. We formally define *visibility segmentation* (Vis) as follows. A reaction r_k to an instruction u_k begins right after the instruction u_k is uttered and ends right before the next instruction u_{k+1} is uttered or right after the IF leaves the area visible at 360° from where u_k was uttered. In the example, instruction 1's reaction would be limited to $\langle 2 \rangle$ because the intersection is not visible from where the instruction was uttered.

The Bhv and Vis methods define how to segment an interaction trace into utterances and their corresponding reactions. However, users frequently perform noisy behavior that is irrelevant to the goal of the task. For example, after hearing an instruction, an IF might go into the wrong room, realize the error, and leave the room. A reaction should not include such irrelevant actions. In addition, IFs may accomplish the same goal using different behaviors: two different IFs may interpret "go to the pink room" by following different paths to the same destination. We would like to be able to generalize both reactions into one canonical reaction.

As a result, our approach *discretizes* reactions into higher-level action sequences with less noise and less variation. Our discretization algorithm uses an *automated planner* and a *planning representation* of the task. This planning representation includes: (1) the task goal, (2) the actions which can be taken in the virtual world, and (3) the current state of the virtual world. Using the planning representation, the planner calculates an optimal path between the starting and ending states of the reaction, eliminating all unnecessary actions. While we use the classical planner FF (Hoffmann, 2003), our technique could also work with *classical planning* (Nau et al., 2004) or other techniques such as *probabilistic planning* (Bonet and Geffner, 2005). It is also not dependent on a particular discretization of the world in terms of actions.

Now we are ready to define *canonical reaction* c_k formally. Let S_k be the state of the virtual world when instruction u_k was uttered, S_{k+1} be the state of the world where the reaction ends (as defined by Bhv or Vis segmentation), and D be the planning domain representation of the virtual world. The *canonical reaction* to u_k is defined as the sequence of actions returned by the planner with S_k as initial state, S_{k+1} as goal state and D as planning domain.

3.2 Interpretation phase

The annotation phase results in a collection of (u_k, c_k) pairs. The interpretation phase uses these pairs to interpret new utterances in three steps. First, we *filter* the set of pairs into those whose reactions can be directly executed from the current IF position. Second, we *group* the filtered pairs according to their reactions. Third, we *select* the group with utterances most similar to the new utterance, and output that group's reaction. Figure 2 shows the output of the first two steps: three groups of pairs whose reactions can all be executed from the IF's current position.

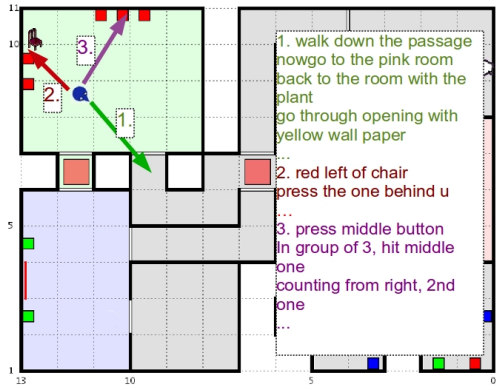


Figure 2: Utterance groups for this situation. Colored arrows show the reaction associated with each group.

We treat the third step, selecting the most similar group for a new utterance, as a classification problem. We compare three different classification methods. One method uses nearest-neighbor classification with three different similarity metrics: Jaccard and Overlap coefficients (both of which measure the degree of overlap between two sets, differing only in the normalization of the final value (Nikravesh et al., 2005)), and Levenshtein Distance (a string metric for measuring the amount of differences between two sequences of words (Levenshtein, 1966)). Our second classification method employs a strategy in which we considered each group as a set of possible machine translations of our utterance, using the BLEU measure (Papineni et al., 2002) to select which group could be considered the best translation of our utterance. Finally, we trained an SVM classifier (Cortes and Vapnik, 1995) using the unigrams

	Corpus C_m		Corpus C_s	
Algorithm	**Bhv**	**Vis**	**Bhv**	**Vis**
Jaccard	47%	54%	54%	70%
Overlap	43%	53%	45%	60%
BLEU	44%	52%	54%	50%
SVM	33%	29%	45%	29%
Levenshtein	21%	20%	8%	17%

Table 1: Accuracy comparison between C_m and C_s for Bhv and Vis segmentation

of each paraphrase and the position of the IF as features, and setting their group as the output class using a libSVM wrapper (Chang and Lin, 2011).

When the system misinterprets an instruction we use a similar approach to what people do in order to overcome misunderstandings. If the system executes an incorrect reaction, the IG can tell the system to cancel its current interpretation and try again using a paraphrase, selecting a different reaction.

4 Evaluation

For the evaluation phase, we annotated both the C_m and C_s corpora entirely, and then we split them in an 80/20 proportion; the first 80% of data collected in each virtual world was used for training, while the remaining 20% was used for testing. For each pair (u_k, c_k) in the testing set, we used our algorithm to predict the reaction to the selected utterance, and then compared this result against the automatically annotated reaction. Table 1 shows the results.

Comparing the Bhv and Vis segmentation strategies, Vis tends to obtain better results than Bhv. In addition, accuracy on the C_s corpus was generally higher than C_m. Given that C_s contained only one IG, we believe this led to less variability in the instructions and less noise in the training data.

We evaluated the impact of user corrections by simulating them using the existing corpus. In case of a wrong response, the algorithm receives a second utterance with the same reaction (a paraphrase of the previous one). Then the new utterance is tested over the same set of possible groups, except for the one which was returned before. If the correct reaction is not predicted after four tries, or there are no utterances with the same reaction, the predictions are registered as wrong. To measure the effects of user corrections vs. without, we used a different evaluation process for this algorithm: first, we split the corpus in a 50/50 proportion, and then we moved correctly predicted utterances from the testing set towards training, until either there was nothing more to learn or the training set reached 80% of the entire corpus size.

As expected, user corrections significantly improve accuracy, as shown in Figure 3. The worst algorithm's results improve linearly with each try, while the best ones behave asymptotically, barely improving after the second try. The best algorithm reaches 92% with just one correction from the IG.

5 Discussion and future work

We presented an approach to instruction interpretation which learns from non-annotated logs of human behavior. Our empirical analysis shows that our best algorithm achieves 70% accuracy on this task, with no manual annotation required. When corrections are added, accuracy goes up to 92% for just one correction. We consider our results promising since state of the art semi-unsupervised approaches to instruction interpretation (Chen and Mooney, 2011) reports a 55% accuracy on manually segmented data.

We plan to compare our system's performance against human performance in comparable situations. Our informal observations of the GIVE corpus indicate that humans often follow instructions incorrectly, so our automated system's performance may be on par with human performance.

Although we have presented our approach in the context of 3D virtual worlds, we believe our technique is also applicable to other domains such as the web, video games, or Human Robot Interaction.

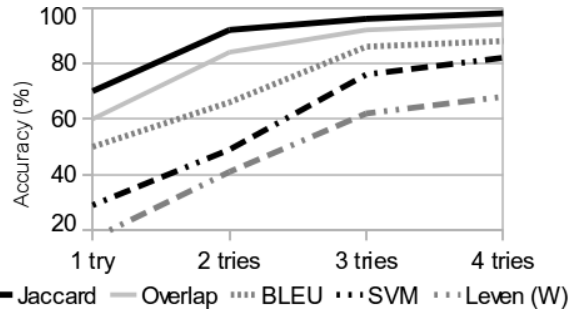


Figure 3: Accuracy values with corrections over C_s

References

Luciana Benotti and Alexandre Denis. 2011. CL system: Giving instructions by corpus based selection. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 296–301, Nancy, France, September. Association for Computational Linguistics.

Blai Bonet and Héctor Geffner. 2005. mGPT: a probabilistic planner based on heuristic search. *Journal of Artificial Intelligence Research*, 24:933–944.

S.R.K. Branavan, Harr Chen, Luke Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. 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 82–90, Suntec, Singapore, August. Association for Computational Linguistics.

Chih-Chung Chang and Chih-Jen 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`.

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-2011)*, pages 859–865, August.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*, 20:273–297.

Myroslava O. Dzikovska, James F. Allen, and Mary D. Swift. 2008. Linking semantic and knowledge representations in a multi-domain dialogue system. *Journal of Logic and Computation*, 18:405–430, June.

Andrew Gargett, Konstantina Garoufi, Alexander Koller, and Kristina Striegnitz. 2010. The GIVE-2 corpus of giving instructions in virtual environments. In *Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC)*, Malta.

James J. Gibson. 1979. *The Ecological Approach to Visual Perception*, volume 40. Houghton Mifflin.

Peter Gorniak and Deb Roy. 2007. Situated language understanding as filtering perceived affordances. *Cognitive Science*, 31(2):197–231.

Jörg Hoffmann. 2003. The Metric-FF planning system: Translating "ignoring delete lists" to numeric state variables. *Journal of Artificial Intelligence Research (JAIR)*, 20:291–341.

Alexander Koller, Kristina Striegnitz, Andrew Gargett, Donna Byron, Justine Cassell, Robert Dale, Johanna Moore, and Jon Oberlander. 2010. Report on the second challenge on generating instructions in virtual environments (GIVE-2). In *Proceedings of the 6th International Natural Language Generation Conference (INLG)*, Dublin.

Tessa Lau, Clemens Drews, and Jeffrey Nichols. 2009. Interpreting written how-to instructions. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 1433–1438, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. Technical Report 8.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: connecting language, knowledge, and action in route instructions. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, pages 1475–1482. AAAI Press.

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 '10, pages 251–258, New York, NY, USA. ACM.

Dana Nau, Malik Ghallab, and Paolo Traverso. 2004. *Automated Planning: Theory & Practice*. Morgan Kaufmann Publishers Inc., California, USA.

Masoud Nikravesh, Tomohiro Takagi, Masanori Tajima, Akiyoshi Shinmura, Ryosuke Ohgaya, Koji Taniguchi, Kazuyosi Kawahara, Kouta Fukano, and Akiko Aizawa. 2005. Soft computing for perception-based decision processing and analysis: Web-based BISC-DSS. In Masoud Nikravesh, Lotfi Zadeh, and Janusz Kacprzyk, editors, *Soft Computing for Information Processing and Analysis*, volume 164 of *Studies in Fuzziness and Soft Computing*, chapter 4, pages 93–188. Springer Berlin / Heidelberg.

Jeff Orkin and Deb Roy. 2009. Automatic learning and generation of social behavior from collective human gameplay. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent SystemsVolume 1*, volume 1, pages 385–392. International Foundation for Autonomous Agents and Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems.

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.

Verena Rieser and Oliver Lemon. 2010. Learning human multimodal dialogue strategies. *Natural Language Engineering*, 16:3–23.

Laura Stoia, Donna K. Byron, Darla Magdalene Shockley, and Eric Fosler-Lussier. 2006. Sentence planning

for realtime navigational instructions. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NAACL-Short '06, pages 157–160, Stroudsburg, PA, USA. Association for Computational Linguistics.

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, pages 806–814, Stroudsburg, PA, USA. Association for Computational Linguistics.

Automatically Mining Question Reformulation Patterns from Search Log Data

Xiaobing Xue*
Univ. of Massachusetts, Amherst
xuexb@cs.umass.edu

Yu Tao*
Univ. of Science and Technology of China
v-yutao@microsoft.com

Daxin Jiang **Hang Li**
Microsoft Research Asia
{djiang,hangli}@microsoft.com

Abstract

Natural language questions have become popular in web search. However, various questions can be formulated to convey the same information need, which poses a great challenge to search systems. In this paper, we automatically mined *5w1h question reformulation patterns* from large scale search log data. The question reformulations generated from these patterns are further incorporated into the retrieval model. Experiments show that using question reformulation patterns can significantly improve the search performance of natural language questions.

1 Introduction

More and more web users tend to use natural language questions as queries for web search. Some commercial natural language search engines such as InQuira and Ask have also been developed to answer this type of queries. One major challenge is that various questions can be formulated for the same information need. Table 1 shows some alternative expressions for the question "how far is it from Boston to Seattle". It is difficult for search systems to achieve satisfactory retrieval performance without considering these alternative expressions.

In this paper, we propose a method of automatically mining *5w1h question*[1] *reformulation patterns* to improve the search relevance of 5w1h questions. *Question reformulations* represent the alternative expressions for 5w1h questions. *A question reformulation pattern* generalizes a set of similar question reformulations that share the same structure. For example, users may ask similar questions "how far is it from X_1 to X_2" where X_1 and X_2 represent some other cities besides Boston and Seattle. Then, similar question reformulations as in Table 1 will be generated with the city names changed. These patterns increase the coverage of the system by handling the queries that did not appear before but share similar structures as previous queries.

Table 1: Alternative expressions for the original question

Original Question:
how far is it from Boston to Seattle
Alternative Expressions:
how many miles is it from Boston to Seattle
distance from Boston to Seattle
Boston to Seattle
how long does it take to drive from Boston to Seattle

Using reformulation patterns as the key concept, we propose a question reformulation framework. First, we mine the question reformulation patterns from search logs that record users' reformulation behavior. Second, given a new question, we use the most relevant reformulation patterns to generate question reformulations and each of the reformulations is associated with its probability. Third, the original question and these question reformulations are then combined together for retrieval.

The contributions of this paper are summarized as two folds. First, we propose a simple yet effective approach to automatically mine 5w1h question reformulation patterns. Second, we conduct comprehensive studies in improving the search performance of 5w1h questions using the mined patterns.

*Contribution during internship at Microsoft Research Asia

[1]5w1h questions start with "Who", "What", "Where", "When", "Why" and "How".

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 187–192,
Jeju, Republic of Korea, 8-14 July 2012.

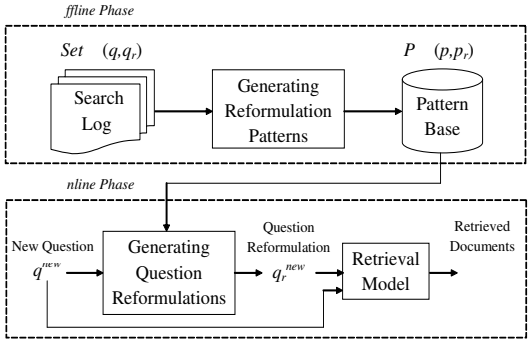


Figure 1: The framework of reformulating questions.

2 Related Work

In the Natural Language Processing (NLP) area, different expressions that convey the same meaning are referred as *paraphrases* (Lin and Pantel, 2001; Barzilay and McKeown, 2001; Pang et al., 2003; Paşca and Dienes, 2005; Bannard and Callison-Burch, 2005; Bhagat and Ravichandran, 2008; Callison-Burch, 2008; Zhao et al., 2008). Paraphrases have been studied in a variety of NLP applications such as machine translation (Kauchak and Barzilay, 2006; Callison-Burch et al., 2006), question answering (Ravichandran and Hovy, 2002) and document summarization (McKeown et al., 2002). Yet, little research has considered improving web search performance using paraphrases.

Query logs have become an important resource for many NLP applications such as class and attribute extraction (Paşca and Van Durme, 2008), paraphrasing (Zhao et al., 2010) and language modeling (Huang et al., 2010). Little research has been conducted to automatically mine 5w1h question reformulation patterns from query logs.

Recently, query reformulation (Boldi et al., 2009; Jansen et al., 2009) has been studied in web search. Different techniques have been developed for query segmentation (Bergsma and Wang, 2007; Tan and Peng, 2008) and query substitution (Jones et al., 2006; Wang and Zhai, 2008). Yet, most previous research focused on keyword queries without considering 5w1h questions.

3 Mining Question Reformulation Patterns for Web Search

Our framework consists of three major components, which is illustrated in Fig. 1.

Table 2: Question reformulation patterns generated for the query pair ("how far is it from Boston to Seattle" ,"distance from Boston to Seattle").

S_1 = {Boston}:("how far is it from X_1 to Seattle" ,"distance from X_1 to Seattle")
S_2 = {Seattle}:("how far is it from Boston to X_1" ,"distance from Boston to X_1")
S_3 = {Boston, Seattle}:("how far is it from X_1 to X_2" ,"distance from X_1 to X_2")

3.1 Generating Reformulation Patterns

From the search log, we extract all successive query pairs issued by the same user within a certain time period where the first query is a 5w1h question. In such query pair, the second query is considered as a question reformulation. Our method takes these query pairs, i.e. $Set = \{(q, q_r)\}$, as the input and outputs a pattern base consisting of 5w1h question reformulation patterns, i.e. $P = \{(p, p_r)\}$). Specifically, for each query pair (q, q_r), we first collect all common words between q and q_r except for stopwords ST[2], where $CW = \{w | w \in q, w \in q', w \notin ST\}$. For any non-empty subset S_i of CW, the words in S_i are replaced as slots in q and q_r to construct a reformulation pattern. Table 2 shows examples of question reformulation patterns. Finally, the patterns observed in many different query pairs are kept. In other words, we rely on the frequency of a pattern to filter noisy patterns. Generating patterns using more NLP features such as the parsing information will be studied in the future work.

3.2 Generating Question Reformulations

We describe how to generate a set of question reformulations $\{q_r^{new}\}$ for an unseen question q^{new}.

First, we search $P = \{(p, p_r)\}$ to find all question reformulation patterns where p matches q^{new}. Then, we pick the best question pattern $p^\star$ according to the number of prefix words and the total number of words in a pattern. We select the pattern that has the most prefix words, since this pattern is more likely to have the same information as q^{new}. If several patterns have the same number of prefix words, we use the total number of words to break the tie.

After picking the best question pattern $p^\star$, we further rank all question reformulation patterns containing $p^\star$, i.e. $(p^\star, p_r)$, according to Eq. 1.

[2]Stopwords refer to the function words that have little meaning by themselves, such as "the", "a", "an", "that" and "those".

Table 3: Examples of the question reformulations and their corresponding reformulation patterns

q^{new}: how good is the eden pure air system $p^\star$: how good is the X		q^{new}: how to market a restaurant $p^\star$: how to market a X	
q_r^{new}	p_r	q_r^{new}	p_r
eden pure air system	X	marketing a restaurant	marketing a X
eden pure air system review	X review	how to promote a restaurant	how to promote a X
eden pure air system reviews	X reviews	how to sell a restaurant	how to sell a X
rate the eden pure air system	rate the X	how to advertise a restaurant	how to advertise a X
reviews on the eden pure air system	reviews on the X	restaurant marketing	X marketing

$$P(p_r|p^\star) = \frac{f(p^\star, p_r)}{\sum_{p'_r} f(p^\star, p'_r)} \quad (1)$$

Finally, we generate k question reformulations q_r^{new} by applying the top k question reformulation patterns containing $p^\star$. The probability $P(p_r|p^\star)$ associated with the pattern $(p^\star, p_r)$ is assigned to the corresponding question reformulation q_r^{new}.

3.3 Retrieval Model

Given the original question q^{new} and k question reformulations $\{q_r^{new}\}$, the query distribution model (Xue and Croft, 2010) (denoted as QDist) is adopted to combine q^{new} and $\{q_r^{new}\}$ using their associated probabilities. The retrieval score of the document D, i.e. $score(q^{new}, D)$, is calculated as follows:

$$score(q^{new}, D) = \lambda \log P(q^{new}|D) + (1-\lambda) \sum_{i=1}^{k} P(p_{r_i}|p^\star) \log P(q_{r_i}^{new}|D) \quad (2)$$

In Eq. 2, λ is a parameter that indicates the probability assigned to the original query. $P(p_{r_i}|p^\star)$ is the probability assigned to $q_{r_i}^{new}$. $P(q_{new}|D)$ and $P(q'|D)$ are calculated using the language model (Ponte and Croft, 1998; Zhai and Lafferty, 2001).

4 Experiments

A large scale search log from a commercial search engine (2011.1-2011.6) is used in experiments. From the search log, we extract all successive query pairs issued by the same user within 30 minutes (Boldi et al., 2008)[3] where the first query is a 5w1h question. Finally, we extracted 6,680,278 question reformulation patterns.

For the retrieval experiments, we randomly sample 10,000 natural language questions as queries from the search log before 2011. For each question, we generate the top ten questions reformulations. The Indri toolkit[4] is used to implement the language model. A web collection from a commercial search engine is used for retrieval experiments. For each question, the relevance judgments are provided by human annotators. The standard NDCG@k is used to measure performance.

Table 4: Retrieval Performance of using question reformulations. * denotes significantly different with Orig.

	NDCG@1	NDCG@3	NDCG@5
Orig	0.2946	0.2923	0.2991
QDist	0.3032*	0.2991*	0.3067*

4.1 Examples and Performance

Table 3 shows examples of the generated questions reformulations. Several interesting expressions are generated to reformulate the original question.

We compare the retrieval performance of using the question reformulations (QDist) with the performance of using the original question (Orig) in Table 4. The parameter λ of QDist is decided using ten-fold cross validation. Two sided t-test are conducted to measure significance.

Table 4 shows that using the question reformulations can significantly improve the retrieval performance of natural language questions. Note that, considering the scale of experiments (10,000 queries), around 3% improvement with respect to NDCG is a very interesting result for web search.

4.2 Analysis

In this subsection, we analyze the results to better understand the effect of question reformulations.

First, we report the performance of always picking the best question reformulation for each query (denoted as Upper) in Table 5, which provides an

[3] In web search, queries issued within 30 minutes are usually considered having the same information need.

[4] www.lemurproject.org/

Table 5: Performance of the upper bound.

	NDCG@1	NDCG@3	NDCG@5
Orig	0.2946	0.2923	0.2991
QDist	0.3032	0.2991	0.3067
Upper	0.3826	0.3588	0.3584

Table 6: Best reformulation within different positions.

top 1	within top 2	within top 3
49.2%	64.7%	75.4%

upper bound for the performance of the question reformulation. Table 5 shows that if we were able to always picking the best question reformulation, the performance of Orig could be improved by around 30% (from 0.2926 to 0.3826 with respect to NDCG@1). It indicates that we do generate some high quality question reformulations.

Table 6 further reports the percent of those 10,000 queries where the best question reformulation can be observed in the top 1 position, within the top 2 positions and within the top 3 positions, respectively.

Table 6 shows that for most queries, our method successfully ranks the best reformulation within the top 3 positions.

Second, we study the effect of different types of question reformulations. We roughly divide the question reformulations generated by our method into five categories as shown in Table 7. For each category, we report the percent of reformulations which performance is bigger/smaller/equal with respect to the original question.

Table 7 shows that the "more specific" reformulations and the "equivalent" reformulations are more likely to improve the original question. Reformulations that make "morphological change" do not have much effect on improving the original question. "More general" and "not relevant" reformulations usually decrease the performance.

Third, we conduct the error analysis on the question reformulations that decrease the performance of the original question. Three typical types of errors are observed. First, some important words are removed from the original question. For example, "what is the role of corporate executives" is reformulated as "corporate executives". Second, the reformulation is too specific. For example, "how to effectively organize your classroom" is reformulated as "how to effectively organize your elementary classroom". Third, some reformulations entirely change the meaning of the original question. For example, "what is the adjective of anxiously" is reformulated as "what is the noun of anxiously".

Table 7: Analysis of different types of reformulations.

Type	increase	decrease	same
Morphological change	11%	10%	79%
Equivalent meaning	32%	30%	38%
More specific/Add words	45%	39%	16%
More general/Remove words	38%	48%	14%
Not relevant	14%	72%	14%

Table 8: Retrieval Performance of other query processing techniques.

	NDCG@1	NDCG@3	NDCG@5
ORIG	0.2720	0.2937	0.3151
NoStop	0.2697	0.2893	0.3112
DropOne	0.2630	0.2888	0.3102
QDist	0.2978	0.3052	0.3250

Fourth, we compare our question reformulation method with two long query processing techniques, i.e. NoStop (Huston and Croft, 2010) and DropOne (Balasubramanian et al., 2010). NoStop removes all stopwords in the query and DropOne learns to drop a single word from the query. The same query set as Balasubramanian et al. (2010) is used. Table 8 reports the retrieval performance of different methods.

Table 8 shows that both NoStop and DropOne perform worse than using the original question, which indicates that the general techniques developed for long queries are not appropriate for natural language questions. On the other hand, our proposed method outperforms all the baselines.

5 Conclusion

Improving the search relevance of natural language questions poses a great challenge for search systems. We propose to automatically mine 5w1h question reformulation patterns from search log data. The effectiveness of the extracted patterns has been shown on web search. These patterns are potentially useful for many other applications, which will be studied in the future work. How to automatically classify the extracted patterns is also an interesting future issue.

Acknowledgments

We would like to thank W. Bruce Croft for his suggestions and discussions.

References

N. Balasubramanian, G. Kumaran, and V.R. Carvalho. 2010. Exploring reductions for long web queries. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 571–578. ACM.

C. Bannard and C. Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics.

R. Barzilay and K.R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 50–57. Association for Computational Linguistics.

S. Bergsma and Q. I. Wang. 2007. Learning noun phrase query segmentation. In *EMNLP-CoNLL07*, pages 819–826, Prague.

R. Bhagat and D. Ravichandran. 2008. Large scale acquisition of paraphrases for learning surface patterns. *Proceedings of ACL-08: HLT*, pages 674–682.

Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, Aristides Gionis, and Sebastiano Vigna. 2008. The query-flow graph: model and applications. In *CIKM08*, pages 609–618.

P. Boldi, F. Bonchi, C. Castillo, and S. Vigna. 2009. From "Dango" to "Japanese Cakes": Query reformulation models and patterns. In *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT'09. IEEE/WIC/ACM International Joint Conferences on*, volume 1, pages 183–190. IEEE.

C. Callison-Burch, P. Koehn, and M. Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24. Association for Computational Linguistics.

C. Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 196–205. Association for Computational Linguistics.

Jian Huang, Jianfeng Gao, Jiangbo Miao, Xiaolong Li, Kuansan Wang, Fritz Behr, and C. Lee Giles. 2010. Exploring web scale language models for search query processing. In *WWW10*, pages 451–460, New York, NY, USA. ACM.

S. Huston and W.B. Croft. 2010. Evaluating verbose query processing techniques. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298. ACM.

B.J. Jansen, D.L. Booth, and A. Spink. 2009. Patterns of query reformulation during web searching. *Journal of the American Society for Information Science and Technology*, 60(7):1358–1371.

R. Jones, B. Rey, O. Madani, and W. Greiner. 2006. Generating query substitutions. In *WWW06*, pages 387–396, Ediburgh, Scotland.

D. Kauchak and R. Barzilay. 2006. Paraphrasing for automatic evaluation.

D.-K. Lin and P. Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Processing*, 7(4):343–360.

K.R. McKeown, R. Barzilay, D. Evans, V. Hatzivassiloglou, J.L. Klavans, A. Nenkova, C. Sable, B. Schiffman, and S. Sigelman. 2002. Tracking and summarizing news on a daily basis with columbia's newsblaster. In *Proceedings of the second international conference on Human Language Technology Research*, pages 280–285. Morgan Kaufmann Publishers Inc.

B. Pang, K. Knight, and D. Marcu. 2003. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 102–109. Association for Computational Linguistics.

M. Paşca and P. Dienes. 2005. Aligning needles in a haystack: Paraphrase acquisition across the web. *Natural Language Processing–IJCNLP 2005*, pages 119–130.

M. Paşca and B. Van Durme. 2008. Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*, pages 19–27.

J. M. Ponte and W. B. Croft. 1998. A language modeling approach to information retrieval. In *SIGIR98*, pages 275–281, Melbourne, Australia.

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. In *ACL02*, pages 41–47.

B. Tan and F. Peng. 2008. Unsupervised query segmentation using generative language models and Wikipedia. In *WWW08*, pages 347–356, Beijing,China.

X. Wang and C. Zhai. 2008. Mining term association patterns from search logs for effective query reformulation. In *CIKM08*, pages 479–488, Napa Valley, CA.

X. Xue and W. B. Croft. 2010. Representing queries as distributions. In *SIGIR10 Workshop on Query Rep-*

resentation and Understanding, pages 9–12, Geneva, Switzerland.

C. Zhai and J. Lafferty. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR01*, pages 334–342, New Orleans, LA.

S. Zhao, H. Wang, T. Liu, and S. Li. 2008. Pivot approach for extracting paraphrase patterns from bilingual corpora. *Proceedings of ACL-08: HLT*, pages 780–788.

S. Zhao, H. Wang, and T. Liu. 2010. Paraphrasing with search engine query logs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1317–1325. Association for Computational Linguistics.

Native Language Detection with Tree Substitution Grammars

Ben Swanson
Brown University
chonger@cs.brown.edu

Eugene Charniak
Brown University
ec@cs.brown.edu

Abstract

We investigate the potential of Tree Substitution Grammars as a source of features for native language detection, the task of inferring an author's native language from text in a different language. We compare two state of the art methods for Tree Substitution Grammar induction and show that features from both methods outperform previous state of the art results at native language detection. Furthermore, we contrast these two induction algorithms and show that the Bayesian approach produces superior classification results with a smaller feature set.

1 Introduction

The correlation between a person's native language (L1) and aspects of their writing in a second language (L2) can be exploited to predict L1 label given L2 text. The International Corpus of Learner English (Granger et al, 2002), or ICLE, is a large set of English student essays annotated with L1 labels that allows us to bring the power of supervised machine learning techniques to bear on this task. In this work we explore the possibility of automatically induced Tree Substitution Grammar (TSG) rules as features for a logistic regression model[1] trained to predict these L1 labels.

Automatic TSG induction is made difficult by the exponential number of possible TSG rules given a corpus. This is an active area of research with two distinct effective solutions. The first uses a nonparametric Bayesian model to handle the large number of rules (Cohn and Blunsom, 2010), while the second is inspired by tree kernel methods and extracts common subtrees from pairs of parse trees (Sangati and Zuidema, 2011). While both are effective, we show that the Bayesian method of TSG induction produces superior features and achieves a new best result at the task of native language detection.

[1] a.k.a. Maximum Entropy Model

2 Related Work

2.1 Native Language Detection

Work in automatic native language detection has been mainly associated with the ICLE, published in 2002. Koppel et al (2005) first constructed such a system with a feature set consisting of function words, POS bi-grams, and character n-grams. These features provide a strong baseline but cannot capture many linguistic phenomena.

More recently, Wong and Dras (2011a) considered syntactic features for this task, using logistic regression with features extracted from parse trees produced by a state of the art statistical parser. They investigated two classes of features: reranking features from the Charniak parser and CFG features. They showed that while reranking features capture long range dependencies in parse trees that CFG rules cannot, they do not produce classification performance superior to simple CFG rules. Their CFG feature approach represents the best performing model to date for the task of native language detection. Wong and Dras (2011b) also investigated the use of LDA topic modeling to produce a latent feature set of reduced dimensionality, but failed to outperform baseline systems with this approach.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 193–197,
Jeju, Republic of Korea, 8-14 July 2012.

2.2 TSG induction

One inherent difficulty in the use of TSGs is controlling the size of grammars automatically induced from data, which with any reasonable corpus quickly becomes too large for modern workstations to handle. When automatically induced TSGs were first proposed by Bod (1991), the problem of grammar induction was tackled with random selection of fragments or weak constraints that led to massive grammars.

A more principled technique is to use a sparse nonparametric prior, as was recently presented by Cohn et al (2009) and Post and Gildea (2009). They provide a local Gibbs sampling algorithm, and Cohn and Blunsom (2010) later developed a block sampling algorithm with better convergence behavior. While this Bayesian method has yet to produce state of the art parsing results, it has achieved state of the art results for unsupervised grammar induction (Blunsom and Cohn, 2010) and has been extended to synchronous grammars for use in sentence compression (Yamangil and Shieber, 2010).

More recently, (Sangati and Zuidema, 2011) presented an elegantly simple heuristic inspired by tree kernels that they call DoubleDOP. They showed that manageable grammar sizes can be obtained from a corpus the size of the Penn Treebank by recording all fragments that occur at least twice, subject to a pairwise constraint of maximality. Using an additional heuristic to provide a distribution over fragments, DoubleDOP achieved the current state of the art for TSG parsing, competing closely with the absolute best results set by refinement based parsers.

2.3 Fragment Based Classification

The use of parse tree fragments for classification began with Collins and Duffy (2001). They used the number of common subtrees between two parse trees as a convolution kernel in a voted perceptron and applied it as a parse reranker. Since then, such tree kernels have been used to perform a variety of text classification tasks, such as semantic role labeling (Moschitti et al, 2008), authorship attribution (Kim et al, 2010), or the work of Suzuki and Isozaki (2006) that performs question classification, subjectivity detection, and polarity identification.

Syntactic features have also been used in non-kernelized classifiers, such as in the work of Wong and Dras (2011a) mentioned in Section 2.1. Additional examples include Raghavan et al (2010), which uses a CFG language model to perform authorship attribution, and Post (2011), which uses TSG features in a logistic regression model to perform grammaticality detection.

3 Tree Substitution Grammars

Tree Substitution Grammars are similar to Context Free Grammars, differing in that they allow rewrite rules of arbitrary parse tree structure with any number of nonterminal or terminal leaves. We adopt the common term *fragment*[2] to refer to these rules, as they are easily visualised as fragments of a complete parse tree.

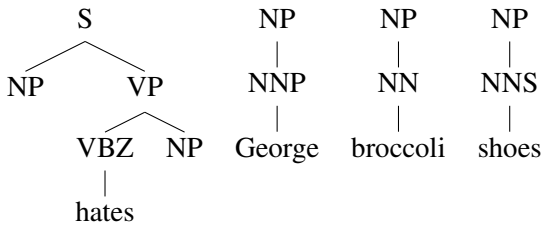


Figure 1: Fragments from a Tree Substitution Grammar capable of deriving the sentences "George hates broccoli" and "George hates shoes".

3.1 Bayesian Induction

Nonparametric Bayesian models can represent distributions of unbounded size with a dynamic parameter set that grows with the size of the training data. One method of TSG induction is to represent a probabilistic TSG with Dirichlet Process priors and sample derivations of a corpus using MCMC.

Under this model the posterior probability of a fragment e is given as

$$P(e|\mathbf{e}^-, \alpha, P_0) = \frac{\#_e + \alpha P_0}{\#_\bullet + \alpha} \qquad (1)$$

where $\mathbf{e}^-$ is the multiset of fragments in the current derivations excluding e, $\#_e$ is the count of the fragment e in $\mathbf{e}^-$, and $\#_\bullet$ is the total number of fragments in $\mathbf{e}^-$ with the same root node as e. P_0 is

[2] As opposed to *elementary tree*, often used in related work

a PCFG distribution over fragments with a bias towards small fragments. α is the concentration parameter of the DP, and can be used to roughly tune the number of fragments that appear in the sampled derivations.

With this posterior distribution the derivations of a corpus can be sampled tree by tree using the block sampling algorithm of Cohn and Blunsom (2010), converging eventually on a sample from the true posterior of all derivations.

3.2 DoubleDOP Induction

DoubleDOP uses a heuristic inspired by tree kernels, which are commonly used to measure similarity between two parse trees by counting the number of fragments they share. DoubleDOP uses the same underlying technique, but caches the shared fragments instead of simply counting them. This yields a set of fragments where each member is guaranteed to appear at least twice in the training set.

In order to avoid unmanageably large grammars only maximal fragments are retained in each pairwise extraction, which is to say that any shared fragment that occurs inside another shared fragment is discarded. The main disadvantage of this method is that the complexity scales quadratically with the training set size, as all pairs of sentences must be considered. It is fully parallelizable, however, which mediates this disadvantage to some extent.

4 Experiments

4.1 Methodology

Our data is drawn from the International Corpus of Learner English (Version 2), which consists of raw unsegmented English text tagged with L1 labels. Our experimental setup follows Wong and Dras (2011a) in analyzing Chinese, Russian, Bulgarian, Japanese, French, Czech, and Spanish L1 essays. As in their work we randomly sample 70 training and 25 test documents for each language. All reported results are averaged over 5 subsamplings of the full data set.

Our data preproccesing pipeline is as follows: First we perform sentence segmentation with OpenNLP and then parse each sentence with a 6 split grammar for the Berkeley Parser (Petrov et al, 2006). We then replace all terminal symbols which do not occur in a list of 598 function words[3] with a single UNK terminal. This aggressive removal of lexical items is standard in this task and mitigates the effect of other unwanted information sources such as topic and geographic location that are correlated with native language in the data.

We contrast three different TSG feature sets in our experiments. First, to provide a baseline, we simply read off the CFG rules from the data set (note that a CFG can be taken as a TSG with all fragments having depth one). Second, in the method we call BTSG, we use the Bayesian induction model with the Dirichlet process' concentration parameters tuned to 100 and run for 1000 iterations of sampling. We take as our resulting finite grammar the fragments that appear in the sampled derivations. Third, we run the parameterless DoubleDOP (2DOP) induction method.

Using the full 2DOP feature set produces over 400k features, which heavily taxes the resources of a single modern workstation. To balance the feature set sizes between 2DOP and BTSG we pass back over the training data and count the actual number of times each fragment recovered by 2DOP appears. We then limit the list to the n most common fragments, where n is the average number of fragments recovered by the BTSG method (around 7k). We refer to results using this trimmed feature set with the label 2DOP, using 2DOP(F) to refer to DoubleDOP with the full set of features.

Given each TSG, we create a binary feature function for each fragment e in the grammar such that the feature $f_e(d)$ is active for a document d if there exists a derivation of some tree $t \in d$ that uses e. Classification is performed with the Mallet package for logistic regression using the default initialized MaxEntTrainer.

5 Results

5.1 Predictive Power

The resulting classification accuracies are shown in Table 1. The BTSG feature set gives the highest performance, and both true TSG induction techniques outperform the CFG baseline.

[3] We use the stop word list distributed with the ROUGE summarization evaluation package.

Model	Accuracy (%)
CFG	72.6
2DOP	73.5
2DOP(F)	76.8
BTSG	78.4

Table 1: Classification accuracy

Model	VoteOne (%)	VoteAll (%)
CFG	69.6	74.7
2DOP	69.1	73.5
BTSG	72.5	76.5

Table 2: Sentence based classification accuracy

The CFG result represents the work of Wong and Dras (2011a), the previous best result for this task. While in their work they report 80% accuracy with the CFG model, this is for a single sampling of the full data set. We observed a large variance in classification accuracy over such samplings, which includes some values in their reported range but with a much lower mean. The numbers we report are from our own implementation of their CFG technique, and all results are averaged over 5 random samplings from the full corpus.

For 2DOP we limit the 2DOP(F) fragments by choosing the 7k with maximum frequency, but there may exist superior methods. Indeed, Wong and Dras (2011a) claims that Information Gain is a better criteria. However, this metric requires a probabilistic formulation of the grammar, which 2DOP does not supply. Instead of experimenting with different limiting metrics, we note that when all 400k rules are used, the averaged accuracy is only 76.8 percent, which still lags behind BTSG.

5.2 Robustness

We also investigated different classification strategies, as binary indicators of fragment occurrence over an entire document may lead to noisy results. Consider a single outlier sentence in a document with a single fragment that is indicative of the incorrect L1 label. Note that it is just as important in the eyes of the classifier as a fragment indicative of the correct label that appears many times. To investigate this phenomena we classified individual sentences, and used these results to vote for each document level label in the test set.

We employed two voting schemes. In the first, VoteOne, each sentence contributes one vote to its maximum probability label. In the second, VoteAll, the probability of each L1 label is contributed as a partial vote. Neither method increases performance for BTSG or 2DOP, but what is more interesting is that in both cases the CFG model outperforms 2DOP (with less than half of the features). The robust behavior of the BTSG method shows that it finds correctly discriminative features across several sentences in each document to a greater extent than other methods.

5.3 Concision

One possible explanation for the superior performance of BTSG is that DDOP is prone to yielding multiple fragments that represent the same linguistic phenomena, leading to sets of highly correlated features. While correlated features are not crippling to a logistic regression model, they add computational complexity without contributing to higher classification accuracy.

To address this hypothesis empirically, we considered pairs of fragments e_A and e_B and calculated the pointwise mutual information (PMI) between events signifying their occurrence in a sentence. For BTSG, the average pointwise mutual information over all pairs (e_A, e_B) is $-.14$, while for 2DOP it is $-.01$. As increasingly negative values of PMI indicate exclusivity, this supports the claim that DDOP's comparative weakness is to some extent due to feature redundancy.

6 Conclusion

In this work we investigate automatically induced TSG fragments as classification features for native language detection. We compare Bayesian and DoubleDOP induced features and find that the former represents the data with less redundancy, is more robust to classification strategy, and gives higher classification accuracy. Additionally, the Bayesian TSG features give a new best result for the task of native language detection.

References

Mohit Bansal and Dan Klein 2010. Simple, accurate parsing with an all-fragments grammar. *Association for Computational Linguistics*.

Phil Blunsom and Trevor Cohn 2010. Unsupervised Induction of Tree Substitution Grammars for Dependency Parsing. *Empirical Methods in Natural Language Processing*.

Rens Bod 1991. A Computational Model of Language Performance: Data Oriented Parsing. *Computational Linguistics in the Netherlands*.

Trevor Cohn, Sharon Goldwater, and Phil Blunsom. 2009. Inducing Compact but Accurate Tree-Substitution Grammars. In *Proceedings NAACL*.

Trevor Cohn, and Phil Blunsom 2010. Blocked inference in Bayesian tree substitution grammars. *Association for Computational Linguistics*.

Michael Collins, Nigel Duffy 2001. Convolution Kernels for Natural Language. *Advances in Neural Information Processing Systems*.

Joshua Goodman 2003. Efficient parsing of DOP with PCFG-reductions. *In Bod et al. chapter 8.*.

S. Granger, E. Dagneaux and F. Meunier. 2002. *International Corpus of Learner English*, (ICLE).

Sangkyum Kim, Hyungsul Kim, Tim Weninger, and Jiawei Han 2010. Authorship classification: a syntactic tree mining approach. *Proceedings of the ACM SIGKDD Workshop on Useful Patterns*.

Koppel, Moshe and Schler, Jonathan and Zigdon, Kfir. 2005. Determining an author's native language by mining a text for errors. *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*.

Alessandro Moschitti, Daniele Pighin and Roberto Basili 2008. Tree Kernels for Semantic Role Labeling. *Computational Linguistics*.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. *Association for Computational Linguistics*.

Matt Post and Daniel Gildea. 2009. Bayesian Learning of a Tree Substitution Grammar. *Association for Computational Linguistics*.

Matt Post. 2011. Judging Grammaticality with Tree Substitution Grammar Derivations. *Association for Computational Linguistics*.

Sindhu Raghavan, Adriana Kovashka and Raymond Mooney 2010. Authorship attribution using probabilistic context-free grammars. *Association for Computational Linguistics*.

Sangati, Federico and Zuidema, Willem 2011. Accurate Parsing with Compact Tree-Substitution Grammars: Double-DOP. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Jun Suzuki and Hideki Isozaki 2006. Sequence and tree kernels with statistical feature mining. *Advances in Neural Information Processing Systems*.

Sze-Meng Jojo Wong and Mark Dras 2011. Exploiting Parse Structures for Native Language Identification. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Sze-Meng Jojo Wong and Mark Dras 2011. Topic Modeling for Native Language Identification. *Proceedings of the Australasian Language Technology Association Workshop*.

Elif Yamangil, Stuart M. Shieber 2010. Bayesian Synchronous Tree-Substitution Grammar Induction and Its Application to Sentence Compression.. *Association for Computational Linguistics*.

Tense and Aspect Error Correction for ESL Learners Using Global Context

Toshikazu Tajiri **Mamoru Komachi** **Yuji Matsumoto**
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
{toshikazu-t, komachi, matsu}@is.naist.jp

Abstract

As the number of learners of English is constantly growing, automatic error correction of ESL learners' writing is an increasingly active area of research. However, most research has mainly focused on errors concerning articles and prepositions even though tense/aspect errors are also important. One of the main reasons why tense/aspect error correction is difficult is that the choice of tense/aspect is highly dependent on global context. Previous research on grammatical error correction typically uses pointwise prediction that performs classification on each word independently, and thus fails to capture the information of neighboring labels. In order to take global information into account, we regard the task as sequence labeling: each verb phrase in a document is labeled with tense/aspect depending on surrounding labels. Our experiments show that the global context makes a moderate contribution to tense/aspect error correction.

1 Introduction

Because of the growing number of learners of English, there is an increasing demand to help learners of English. It is highly effective for learners to receive feedback on their essays from a human tutor (Nagata and Nakatani, 2010). However, manual feedback needs a lot of work and time, and it also requires much grammatical knowledge. Thus, a variety of automatic methods for helping English learning and education have been proposed.

The mainstream of English error detection and correction has focused on article errors (Knight and Chander, 1994; Brockett et al., 2006) and preposition errors (Chodorow et al., 2007; Rozovskaya and Roth, 2011), that commonly occur in essays by ESL learners. On the other hand, tense and aspect errors have been little studied, even though they are also commonly found in learners' essays (Lee and Seneff, 2006; Bitchener et al., 2005). For instance, Lee (2008) corrects English verb inflection errors, but they do not deal with tense/aspect errors because the choice of tense and aspect highly depends on global context, which makes correction difficult. Consider the following sentences taken from a corpus of a Japanese learner of English.

(1) I had a good time this Summer Vacation.
First, I *go to KAIYUKAN [1] with my friends.

In this example, *go* in the second sentence should be written as *went*. It is difficult to correct this type of error because there are two choices for correction, namely *went* and *will go*. In this case, we can exploit global context to determine which correction is appropriate: the first sentence describes a past event, and the second sentence refers the first sentence. Thus, the verb should be changed to past tense. This deduction is easy for humans, but is difficult for machines.

One way to incorporate such global context into tense/aspect error correction is to use a machine learning-based sequence labeling approach. Therefore, we regard the task as sequence labeling: each verb phrase in the document is labeled with tense/aspect depending on surrounding labels. This model naturally takes global context into account. Our experiments show that global context makes a moderate contribution to tense/aspect correction.

[1] Kaiyukan is an aquarium in Osaka, Japan.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 198–202,
Jeju, Republic of Korea, 8-14 July 2012.

2 Tense/Aspect Error Corpus

Developing a high-quality tense and aspect error correction system requires a large corpus annotated with tense/aspect errors. However, existing annotated corpora are limited in size,[2] which precludes the possibility of machine learning-based approach. Therefore, we constructed a large-scale tense/aspect corpus from *Lang-8*,[3] a social networking service for learners of foreign languages. ESL learners post their writing to be collaboratively corrected by native speakers. We leverage these corrections in creating our tense/aspect annotation. Lang-8 has 300,000 users from 180 countries worldwide, with more than 580,000 entries, approximately 170,000 of them in English.[4] After cleaning the data, the corpus consists of approximately 120,000 English entries containing 2,000,000 verb phrases with 750,000 verb phrases having corrections.[5] The annotated tense/aspect labels include 12 combinations of tense (past, present, future) and aspect (nothing, perfect, progressive, perfect progressive).

3 Error Correction Using Global Context

As we described in Section 1, using only local information about the target verb phrase may lead to inaccurate correction of tense/aspect errors. Thus, we take into account global context: the relation between target and preceding/following verb phrases. In this paper, we formulate the task as sequence labeling, and use Conditional Random Fields (Lafferty, 2001), which provides state-of-the-art performance in sequence labeling while allowing flexible feature design for combining local and global feature sets.

3.1 Local Features

Table 1 shows the local features used to train the error correction model.

Table 1: Local features for a verb phrase

name	description
t-learn	tense/aspect written by the learner (surface tense/aspect)
bare	the verb lemma
L	the word to the left
R	the word to the right
nsubj	nominal subject
dobj	direct object
aux	auxiliary verb
pobj	object of a preposition
p-tmod	temporal adverb
norm-p-tmod	normalized temporal adverb
advmod	other adverb
conj	subordinating conjunction
main-clause	true if the target VP is in main clause
sub-clause	true if the target VP is in subordinate clause

We use dependency relations such as **nsubj**, **dobj**, **aux**, **pobj**, and **advmod** for syntactic features. If a sentence including a target verb phrase is a complex sentence, we use the **conj** feature and add either the **main-clause** or the **sub-clause** feature depending on whether the target verb is in the main clause or in a subordinate clause. For example, the following two sentences have the same features although they have different structures.

(2) It pours when it rains.

(3) When it rains it pours.

In both sentences, we use the feature **main-clause** for the verb phrase *pours*, and **sub-clause** for the verb phrase *rains* along with the feature **conj:when** for both verb phrases.

Regarding **p-tmod**, we extract a noun phrase including a word labeled **tmod** (temporal adverb). For instance, consider the following sentence containing a temporal adverb:

(4) I had a good time last night.

In (4), the word *night* is the head of the noun phrase *last night* and is a temporal noun,[6] so we add the feature **p-tmod:last night** for the verb phrase *had*.

Additionally, **norm-p-tmod** is a normalized form of **p-tmod**. Table 2 shows the value of the feature **norm-p-tmod** and the corresponding temporal keywords. We use **norm-p-tmod** when **p-tmod**

[2]Konan-JIEM Learner Corpus Second Edition (`http://gsk.or.jp/catalog/GSK2011-B/catalog.html`) contains 170 essays, and Cambridge English First Certificate in English (`http://www.cambridgeesol.org/exams/fce/index.html`) contains 1244 essays.

[3]`http://lang-8.com/`

[4]As of January, 2012. More details about the Lang-8 corpus can be found in (Mizumoto et al., 2011).

[5]Note that not all the 750,000 verb phrases were corrected due to the misuse of tense/aspect.

[6]We made our own temporal noun list.

Table 2: The value of the feature **norm-p-tmod** and corresponding temporal keywords

temporal keywords	value
yesterday or *last*	past
now	present
tomorrow or *next*	future
today or *this*	this

Table 3: Feature templates

Local Feature Templates
<head> <head, t-learn> <head, L, R> <L> <L, head>
<L, t-learn> <R> <R, head> <R, t-learn> <nsubj>
<nsubj, t-learn> <aux> <aux, head> <aux, t-learn>
<pobj> <pobj, t-learn> <norm-p-tmod>
<norm-p-tmod, t-learn> <advmod> <advmod, t-learn>
<tmod> <tmod, t-learn> <conj> <conj, t-learn>
<main-clause> <main-clause, t-learn>
<sub-clause> <sub-clause, t-learn>
<conj, main-clause> <conj, sub-clause>
Global Context Feature Templates
<p-tmod′> <p-tmod′, t-learn> <p-tmod′, t-learn′>
<p-tmod′, t-learn′, t-learn> <norm-p-tmod′>
<norm-p-tmod′, t-learn> <norm-p-tmod′, t-learn′>
<norm-p-tmod′, t-learn′, t-learn>

includes any temporal keywords. For instance, in the sentence (4), we identify *last night* as temporal adverb representing past, and thus create a feature **time:past** for the verb phrase *had*.

3.2 Feature Template

Table 3 shows feature templates. <*a*> represents a singleton feature and <*a, b*> represents a combination of features *a* and *b*. Also, *a′* means the feature *a* of the preceding verb phrase. A local feature template is a feature function combining features in the target verb phrase, and a global context feature template is a feature function including features from a non-target verb phrase. Suppose we have following learner's sentences:

(5) I **went** to Kyoto yesterday.
I ***eat** yatsuhashi[7] and **drank** green tea.

In (5), the verb before *eat* is *went*, and **p-tmod:yesterday** and **norm-p-tmod:past** are added to the feature set of verb *went*. Accordingly, the global context features **p-tmod′:yesterday** and **norm-p-tmod′:past** are added to the verb *eat*.

[7] Yatsuhashi is a Japanese snack.

Table 4: Example of global context feature functions generated by feature templates

<p-tmod′:yesterday>
<p-tmod′:yesterday, t-learn′:simple past>
<p-tmod′:yesterday, t-learn:simple present>
<p-tmod′:yesterday, t-learn′:simple past, t-learn:simple past>
<norm-p-tmod′:past>
<norm-p-tmod′:past, t-learn′:simple past>
<norm-p-tmod′:past, t-learn:simple present>
<norm-p-tmod′:past, t-learn′:simple past, t-learn:simple present>

Table 4 lists all the global context features for the verb *eat* generated by the feature templates.

3.3 Trade-off between Precision and Recall

Use of surface tense/aspect forms of target verbs improves precision but harms recall. This is because in most cases the surface tense/aspect and the correct tense/aspect form of a verb are the same. It is, of course, desirable to achieve high precision, but very low recall leads to the system making no corrections. In order to control the trade-off between precision and recall, we re-estimate the best output label $\hat{y}$ based on the originally estimated label y as follows:

$$\hat{y} = \arg\max_{y} s(y)$$

$$s(y) = \begin{cases} \alpha c(y), & \text{if y is the same as learner's tense/aspect} \\ c(y) & \text{otherwise.} \end{cases}$$

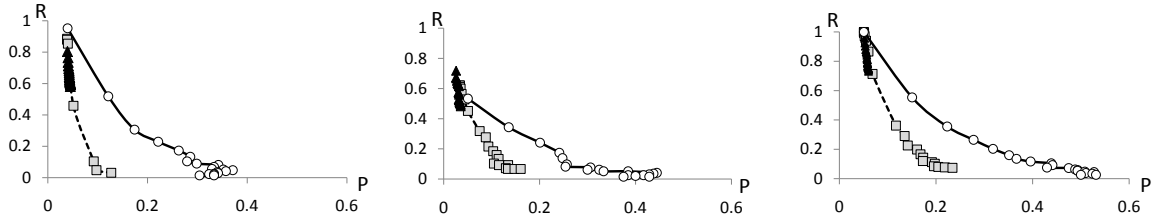
where $c(y)$ is the confidence value of y estimated by the originally trained model (explained in 4.3), and α ($0 \leq \alpha < 1$) is the weight of the surface tense/aspect.

We first calculate $c(y)$ of all the labels, and discount only the label that is the same as learner's tense/aspect, and finally we choose the best output label. This process leads to an increase of recall. We call this method **T-correction**.

4 Experiments

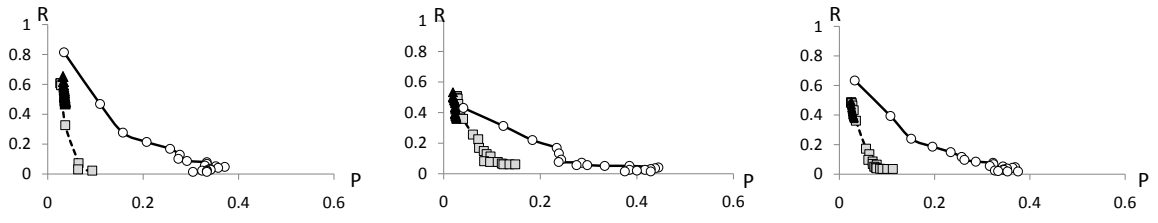
4.1 Data and Feature Extraction

We used the Lang-8 tense/aspect corpus described in Section 2. We randomly selected 100,000 entries for training and 1,000 entries for testing. The test



(a) tense (b) aspect (c) tense/aspect

Figure 1: Precision-Recall curve for error detection



(a) tense (b) aspect (c) tense/aspect

Figure 2: Precision-Recall curve for error correction

SVM MAXENT CRF

data includes 16,308 verb phrases, of which 1,072 (6.6%) contain tense/aspect errors. We used Stanford Parser 1.6.9 [8] for generating syntactic features and tense/aspect tagging.

4.2 Classifiers

Because we want to know the effect of using global context information with CRF, we trained a one-versus-rest multiclass SVM and a maximum entropy classifier (MAXENT) as baselines.

We built a SVM model with LIBLINEAR 1.8[9] and a CRF and a MAXENT model with CRF++ 0.54.[10] We use the default parameters for each toolkit.

In every method, we use the same features and feature described in Section 3, and use T-correction for choosing the final output. The confidence measure of the SVM is the distance to the separating hyperplane, and that of the MAXENT and the CRF is the marginal probability of the estimated label.

[8] http://nlp.stanford.edu/software/lex-parser.shtml

[9] http://www.csie.ntu.edu.tw/~cjlin/liblinear/

[10] http://crfpp.sourceforge.net/

5 Results

Figures 1 and 2 show the Precision-Recall curves of the error detection and correction performance of each model. The figures are grouped by error types: tense, aspect, and both tense and aspect. All figures indicate that the CRF model achieves better performance than SVM and MAXENT.

6 Analysis

We analysed the results of experiments with the α parameter of the CRF model set to 0.1. The most frequent type of error in the corpus is using simple present tense instread of simple past, with 211 instances. Of these our system detected 61 and successfully corrected 52 instances. However, of the second most frequent error type (using simple past instead of simple present), with 94 instances in the corpus, our system only detected 9 instances. One reason why the proposed method achieves high performance in the first type of errors is that tense errors with action verbs written as simple present are relatively easy to detect.

References

John Bitchener, Stuart Young, and Denise Cameron. 2005. The Effect of Different Types of Corrective Feedback on ESL Student Writing. *Journal of Second Language Writing*, 14(3):191–205.

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL Errors Using Phrasal SMT Techniques. In *Proceedings of COLING-ACL*, pages 249–256.

Martin Chodorow, Joel R. Tetreault, and Na-Rae Han. 2007. Detection of Grammatical Errors Involving Prepositions. In *Proceedings of ACL-SIGSEM*, pages 25–30.

Kevin Knight and Ishwar Chander. 1994. Automated Postediting of Documents. In *Proceedings of the AAAI'94*, pages 779–784.

John Lafferty. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of ICML*, pages 282–289.

John Lee and Stephanie Seneff. 2006. Automatic Grammar Correction for Second-Language Learners. In *Proceedings of the 9th ICSLP*, pages 1978–1981.

John Lee and Stephanie Seneff. 2008. Correcting Misuse of Verb Forms. In *Proceedings of the 46th ACL:HLT*, pages 174–182.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining Revision Log of Language Learning SNS for Automated Japanese Error Correction of Second Language Learners. In *Proceedings of 5th IJCNLP*, pages 147–155.

Ryo Nagata and Kazuhide Nakatani. 2010. Evaluating Performance of Grammatical Error Detection to Maximize Learning Effect. In *Proceedings of COLING*, pages 894–900.

Alla Rozovskaya and Dan Roth. 2011. Algorithm Selection and Model Adaptation for ESL Correction Tasks. In *Proceedings of the 49th ACL:HLT*, pages 924–933.

Movie-DiC: a Movie Dialogue Corpus for Research and Development

Rafael E. Banchs
Human Language Technology
Institute for Infocomm Research
Singapore 138632
rembanchs@i2r.a-star.edu.sg

Abstract

This paper describes Movie-DiC a Movie Dialogue Corpus recently collected for research and development purposes. The collected dataset comprises 132,229 dialogues containing a total of 764,146 turns that have been extracted from 753 movies. Details on how the data collection has been created and how it is structured are provided along with its main statistics and characteristics.

1 Introduction

Data driven applications have proliferated in Computational Linguistics during the last decade. Several factors, such as the availability of more powerful computers, an almost unlimited storage capacity, the availability of large volumes of data in digital format, as well as the recent advances in machine learning theory, have significantly contributed to such a proliferation.

Among the many applications that have benefited from this data-driven boom, probably the most representative examples are: information retrieval (Qin *et al.*, 2008), machine translation (Brown *et al.*, 1993), question answering (Molla-Aliod and Vicedo, 2010) and dialogue systems (Rieser and Lemon, 2011).

In the specific case of dialogue systems, data acquisition can impose some challenges depending on the specific domain and task the dialogue system is targeted for. In some specific domains, in which human-human dialogue applications already exists, data collection is generally straight forward, while in some other cases, data design and collection can constitute a complex problem (Williams and Young, 2003; Zue, 2007; Misu *et al.*, 2009).

Depending on the objective being pursued, dialogue systems can be grouped into two major categories: task-oriented and chat-oriented systems. In the first case, the system is required to help the user to accomplish a specific goal or objective (Busemann *et al.*, 1997; Stallard, 2000). In the second case, the system objective is mainly entertainment oriented. Systems in this category are required to play, chitchat or just accompany the user (Weizenbaum, 1966; Wallis, 2010).

In this work, we focus our attention on dialogue data which is suitable for training chat-oriented dialogue systems. Different from task-oriented dialogue collections (Mann, 2003), instead of being concentrated on a specific domain or area of knowledge, the training dataset for a chat-oriented dialogue system must cover a wide variety of domains, as well as be able to provide a fair representation of world-knowledge semantics and pragmatics (Bunt, 2000). To this end, we have collected dialogues from movie scripts aiming at constructing a dialogue corpus which should provide a good sample of domains, styles and world knowledge, as well as constitute a valuable resource for research and development purposes.

The rest of the paper is structured as follows. Section 2 describes in detail the implemented collection process and the structure of the generated database. Section 3 presents the main statistics, as well as the main characteristics of the resulting corpus. Finally, section 4 presents our conclusions and future work plans.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 203–207,
Jeju, Republic of Korea, 8-14 July 2012.

2 Collecting Dialogues from Movies

As already stated in the introduction, our presented dialogue corpus has been extracted from movie scripts. More specifically, scripts freely available from The Internet Movie Script Data Collection (http://www.imsdb.com/) have been used. In this section we describe the implemented data collection process and the data structure finally used for the generated corpus.

As a first step of the collection construction, dialogues have to be identified and extracted from the crawled html files. Three basic types of information elements are extracted from the scripts: speakers, utterances and context.

The utterance and speaker information elements contain what is said at each dialogue turn and the corresponding character who says it, respectively. Context information elements, on the other hand, contain all additional information/texts appearing in the scripts, which are typically of narrative nature and explain what is happening in the scene.

Figure 1 depicts a browser snapshot illustrating the typical layout of a movie script and the most common spatial distribution of the aforementioned information elements.

It is important to mention that a lot of different variants to the format presented in Figure 1 can be actually encountered in The Internet Movie Script Data Collection. Because of this, our parsing algorithms had to be revised and adjusted several times in order to achieve a reasonable level of robustness that allowed for processing the largest possible amount of movie scripts.

Another important problem was the identification of dialogue boundaries. Some heuristics were implemented by taking into account the size and number of context elements between speaker turns.

A post-processing step was also implemented to either filter out or amend some of the most common parsing errors occurring during the extraction phase. Some of these errors include: corrupted formats, turn continuations, notes inserted within the turn, misspelling of speaker names, etc.

In addition to this, a semi-automatic process was still necessary to filter out movie scripts exhibiting extremely different layouts or invalid file formats. Approximately, 17% of the movie scripts crawled from The Internet Movie Script Data Collection had to be discarded. From a total of 911 crawled scripts, only 753 were successfully processed.

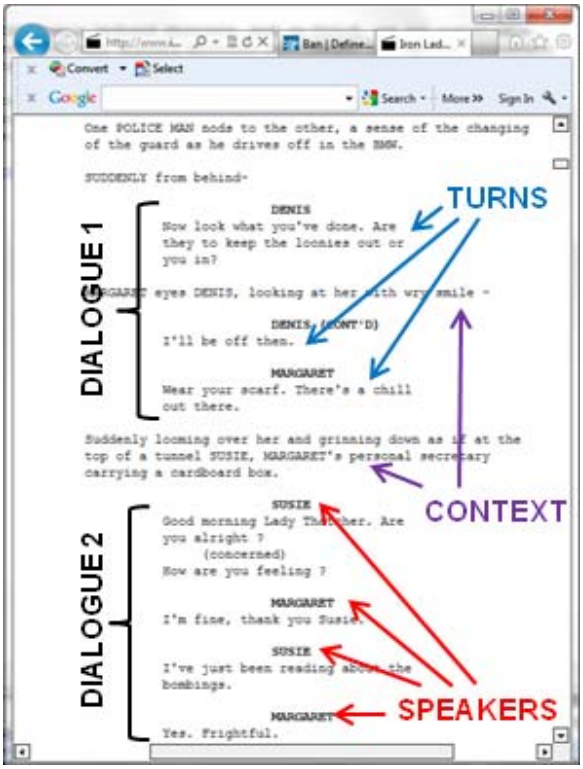


Figure 1: Typical layout of a movie script

The extracted information was finally organized in dialogical units, in which the information regarding turn sequences inside each dialogue, as well as dialogue sequences within each movie script was preserved. Figure 2 illustrates an example of the XML representation for one of the dialogues extracted from *Who Framed Roger Rabbit.*

```
<dialogue id="47" n_utterances="4">
   <speaker>VALIANT</speaker>
      <context></context>
      <utterance>You shot Roger.</utterance>
   <speaker>JESSICA RABBIT</speaker>
      <context>Jessica moves the box aside and
tugs on the rabbit ears. The rabbit head pops
off. Underneath is a Weasle. In his hand is the
Colt .45 Buntline.</context>
      <utterance>That's not Roger. It's one of
Doom's men. He killed R.K. Maroon.</utterance>
   <speaker>VALIANT</speaker>
      <context></context>
      <utterance>Lady, I guess I had you pegged
wrong.</utterance>
   <speaker>JESSICA RABBIT</speaker>
      <context>As they run down the
alley...</context>
      <utterance>Don't worry, you're not the
first. We better get out of here.</utterance>
</dialogue>
```

Figure 2: An example of a dialogue unit

3 Movie Dialogue Corpus Statistics

In this section we present the main statistics of the resulting dialogue corpus and study some of its more important properties. The final dialogue collection was the result of successfully processing 753 movie scripts. Table 1 summarizes the main statistics of the resulting dialogue collection.

Total number of scripts collected	911
Total number of scripts processed	753
Total number of dialogues	132,229
Total number of speaker turns	764,146
Average amount of dialogues per movie	175.60
Average amount of turns per movie	1,014.80
Average amount of turns per dialogue	5.78

Table 1: Main statistics of the collected movie dialogue dataset

Movies were mainly crawled from the action, crime, drama and thriller genres. However, as each movie commonly belongs to more than one single genre, much more genres are actually represented in the dataset. Table 2 summarizes the distribution of movies by genre (notice that, as most of the movies belong to more than one genre, the total summation of percentages exceeds 100%).

Genre	**Movies**	**Percentage**
Action	258	34.26
Adventure	133	17.66
Animation	22	2.92
Comedy	149	19.79
Crime	163	21.65
Drama	456	60.56
Family	31	4.12
Fantasy	82	10.89
Horror	104	13.81
Musical	18	2.39
Mystery	95	12.62
Romance	123	16.33
Sci-Fi	129	17.13
Thriller	329	43.69
War	25	3.32
Western	11	1.46

Table 2: Distribution of movies per genre

The first characteristic of the corpus to be analyzed is the distribution of dialogues per movie. This distribution is shown in Figure 3. As seen from the figure, the distribution of dialogues per movie is clearly symmetric around its mean value of 175 dialogues per movie. For most of the movies in the collection, a number of dialogues ranging from about 100 to 250 were extracted.

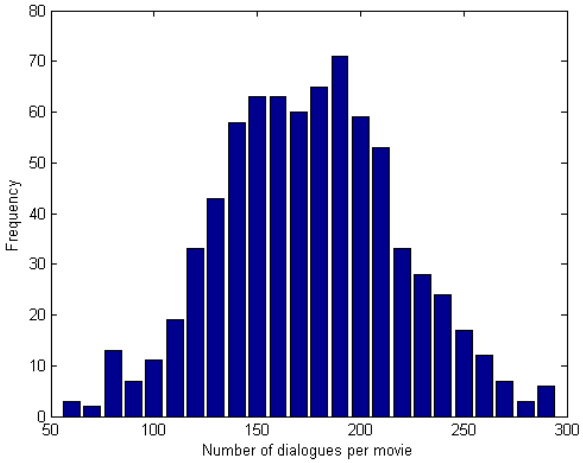


Figure 3: Distribution of dialogues per movie

The second property of the corpus to be studied is the distribution of turns per dialogue. This distribution is shown in Figure 4. As seen from the figure, this distribution approximates a power law behavior, with a large number of very short dialogues (about 50K two-turn dialogues) and a small amount of long dialogues (only six dialogues with more than 200 turns). The median of the distribution is 5.63 turns per dialogue.

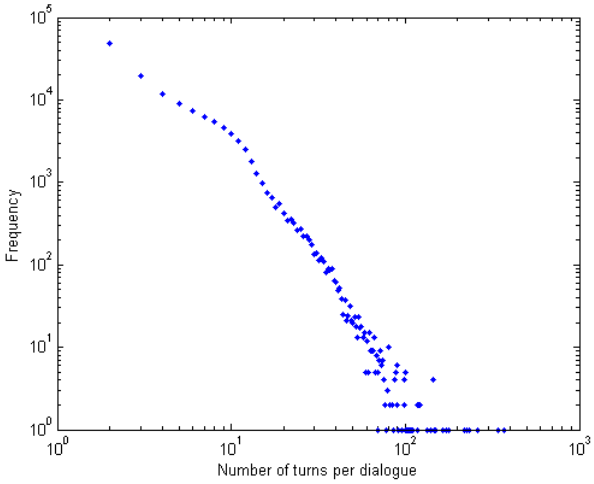


Figure 4: Distribution of turns per dialogue

The third property of the corpus to be described is the distribution of number of speakers per dia-

logue. This distribution is shown in Figure 5. As seen from the bar-plot depicted in the figure, the largest proportion of dialogues (around 60K) involves two speakers. The second largest proportion of "dialogues" (about 35K) involves only a single speaker, which means that this subset of the data collection is actually composed by monologues or single speaker interventions. The third and fourth larger proportions are those involving three and four speakers, respectively.

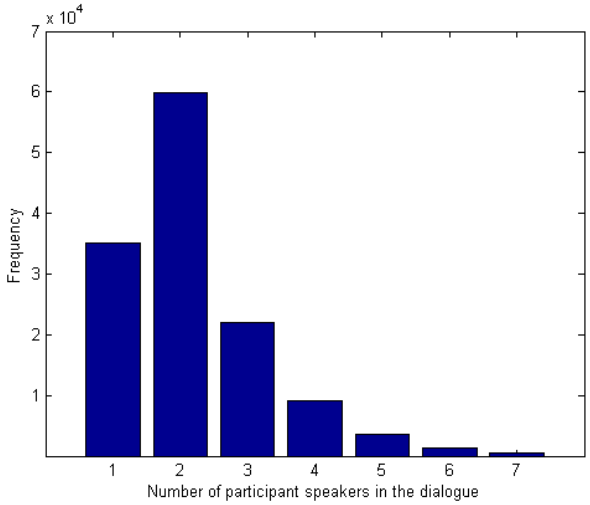


Figure 5: Distribution of number of speakers per dialogue

Finally, in Figure 6, we present a cross-plot between the number of dialogues and the number of turns within each movie script.

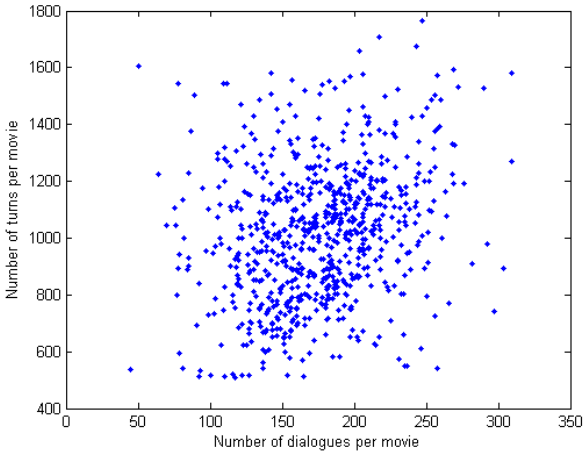


Figure 6: Cross-plot between the number of dialogues and turns within each movie script

As seen from the cross-plot, an average movie has between 150 and 200 dialogues comprising between 1000 and 1200 turns in total. The cross-plot also reveals some interesting extreme cases in the data collection.

For instance, movies with few dialogues but many turns are located towards the upper-left corner of the figure. In this zone we can find movies as: *Happy Birthday Wanda June*, *Hannah and Her Sisters* and *All About Eve*. In the lower-left corner of the figure we can find movies with few dialogues and few turns, as for instance: *1492 Conquest of Paradise* and *The Cooler*.

In the right side of the figure we find the lots-of-dialogues region. There we can find movies with lots of very short dialogues (lower-right corner), such as *Jimmy and Judy* and *Walking Tall*; or movies with lots of dialogues and turns (upper-right corner), such as *The Curious Case of Benjamin Button* and *Jennifer's Body*.

4 Conclusions and Future Work

In this paper, we have described Movie-DiC a Movie Dialogue Corpus that has been collected for research and development purposes. The data collection comprises 132,229 dialogues containing a total of 764,146 turns/utterances that have been extracted from 753 movies. Details on how the data collection has been created and how the corpus is structured were provided along with the main statistics and characteristics of the corpus.

Although strictly speaking, and by its particular nature, Movie-DiC does not constitute a corpus of real human-to-human dialogues, it does constitute an excellent dataset for studying the semantic and pragmatic aspects of human communication within a wide variety of contexts, scenarios, styles and socio-cultural settings.

Specific technologies and applications that can exploit a resource like this include, but are not restricted to: example-based chat bots (Banchs and Li, 2012), question answering systems, discourse and pragmatics analysis, narrative vs. colloquial style classification, genre classification, etc.

As future work, we intend to expand the current size of the collection from 0.7K to 2K movies, as well as to improve some of our parsing and post-processing algorithms for reducing the amount of noise still present in the collection and enhance the quality of the current version of the dataset.

Acknowledgments

The author would like to thank the Institute for Infocomm Research for its support and permission to publish this work.

References

Banchs R E, Li H (2012) IRIS: a chat-oriented dialogue system based on the vector space model. In Proceedings of the 50th Annual Meeting of the ACL, demo session.

Brown P, Della Pietra S, Della Pietra V, Mercer R (1993) The mathematics of statistical machine translation: parameter estimation. Computational Linguistics 19(2):263-311.

Bunt H (ed) (2000) Abduction, belief, and context in dialogue: studies in computational pragmatics. J. Benjamins.

Busemann S, Declerck T, Diagne A, Dini L, Klein J, Schmeier S (1997) Natural language dialogue service for appointment scheduling agents. In Proceedings of the 5th Conference on Applied Natural Language Processing, pp 25-32.

Mann W (2003) The Dialogue Diversity Corpus. Accessed online on 16 March 2012 from: http://www-bcf.usc.edu/~billmann/diversity/DDivers-site.htm

Misu T, Ohtake K, Hori C, Kashioka H, Nakamura S (2009) Annotating communicative function and semantic content in dialogue act for construction of consulting dialogue systems. In Proceedings of the Int. Conf. of Spoken Language Processing

Molla-Aliod D, Vicedo J (2010) Question answering. In Indurkhya and Damerau (eds) Handbook of Natural Language Processing, pp 485-510. Chapman & Hall.

Qin T, Liu T, Zhang X, Wang D, Xiong W, Li H (2008) Learning to rank relational objects and its application to Web search. In Proceedings of the 17th International Conference on World Wide Web, pp 407-416.

Rieser V, Lemon O (2011) Reinforcement learning for adaptive dialogue systems: a data-driven methodology for dialogue management and natural language generation. Springer.

Stallard D (2000) Talk'n'travel: a conversational system for air travel planning. In Proceedings of the 6th Conference on Applied Natural Language Processing, pp 68-75.

Wallis P (2010) A robot in the kitchen. In Proceedings of the ACL 2010 Workshop on Companionable Dialogue Systems, pp 25-30.

Weizenbaum J (1966) ELIZA – A computer program for the study of natural language communication between man and machine. Communications of the ACM 9(1):36-45.

Williams J, Young S (2003) Using Wizard-of-Oz simulations to bootstrap Reinforcement-Learning-based dialog management systems. In Proceedings of the 4th SIGDIAL Workshop on Discourse and Dialogue.

Zue V (2007) On organic interfaces. In Proceedings of the International Conference of Spoken Language Processing.

Combining Textual Entailment and Argumentation Theory for Supporting Online Debates Interactions

Elena Cabrio and **Serena Villata**
INRIA
2004 Route des Lucioles BP93
06902 Sophia-Antipolis cedex, France.
{elena.cabrio, serena.villata}@inria.fr

Abstract

Blogs and forums are widely adopted by online communities to debate about various issues. However, a user that wants to cut in on a debate may experience some difficulties in extracting the current accepted positions, and can be discouraged from interacting through these applications. In our paper, we combine textual entailment with argumentation theory to automatically extract the arguments from debates and to evaluate their acceptability.

1 Introduction

Online debate platforms, like Debatepedia[1], Twitter[2] and many others, are becoming more and more popular on the Web. In such applications, users are asked to provide their own opinions about selected issues. However, it may happen that the debates become rather complicated, with several arguments supporting and contradicting each others. Thus, it is difficult for potential participants to understand the way the debate is going on, i.e., which are the current accepted arguments in a debate. In this paper, we propose to support participants of online debates with a framework combining Textual Entailment (TE) (Dagan et al., 2009) and abstract argumentation theory (Dung, 1995). In particular, TE is adopted to extract the abstract arguments from natural language debates and to provide the relations among these arguments; argumentation theory is then used to compute the set of accepted arguments among those obtained from the TE module, i.e., the arguments shared by the majority of the participants without being attacked by other accepted arguments. The originality of the proposed framework lies in the combination of two existing approaches with the goal of supporting participants in their interactions with online debates, by automatically detecting the arguments in natural language text, and identifying the accepted ones. We evaluate the feasibility of our combined approach on a set of arguments extracted from a sample of Debatepedia.

[1] http://debatepedia.idebate.org
[2] http://twitter.com/

2 First step: textual entailment

TE was proposed as an applied framework to capture major semantic inference needs across applications in NLP, e.g. (Romano et al., 2006; Barzilay and McKeown, 2005; Nielsen et al., 2009). It is defined as a relation between two textual fragments, i.e., the text (T) and the hypothesis (H). Entailment holds if the meaning of H can be inferred from the meaning of T, as interpreted by a typical language user. Consider the pairs in Example 1 and 2.

Example 1.
T1: *Research shows that drivers speaking on a mobile phone have much slower reactions in braking tests than non-users, and are worse even than if they have been drinking.*
H:*The use of cell-phones while driving is a public hazard.*

Example 2 (Continued)**.**
T2: *Regulation could negate the safety benefits of having a phone in the car. When you're stuck in traffic, calling to say you'll be late can reduce stress and make you less inclined to drive aggressively to make up lost time.*
H:*The use of cell-phones while driving is a public hazard.*

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 208–212,
Jeju, Republic of Korea, 8-14 July 2012.

A system aimed at recognizing TE should detect an entailment relation between T1 and H (Example 1), and a contradiction between T2 and H (Example 2).

As introduced before, our paper proposes an approach to support the participants in forums or debates to detect the accepted arguments among those expressed by the other participants on a certain topic. As a first step, we need to *(i)* automatically recognize a participant's opinion on a certain topic as an argument, as well as to *(ii)* detect its relationship with the other arguments. We therefore cast the described problem as a TE problem, where the T-H pair is a pair of arguments expressed by two different participants on a certain topic. For instance, given the argument "The use of cell-phones while driving is a public hazard" (that we consider as H as a starting point), participants can support it expressing arguments from which H can be inferred (Example 1), or can contradict such argument with opinions against it (Example 2). Since in debates arguments come one after the other, we extract and compare them both with respect to the main issue, and with the other participants' arguments (when the new argument entails or contradicts one of the arguments previously expressed by another participant). For instance, given the same debate as before, a new argument T3 may be expressed by a third participant with the goal of contradicting T2 (that becomes the new H (H1) in the pair), as shown in Example 3.

Example 3 (Continued)**.**
T3: *If one is late, there is little difference in apologizing while in their car over a cell phone and apologizing in front of their boss at the office. So, they should have the restraint to drive at the speed limit, arriving late, and being willing to apologize then; an apologetic cell phone call in a car to a boss shouldn't be the cause of one being able to then relax, slow-down, and drive the speed-limit.*
T2 → H1: *Regulation could negate the safety benefits of having a phone in the car. When you're stuck in [...]*

TE provides us with the techniques to detect both the arguments in a debate, and the kind of relation underlying each couple of arguments. The TE system returns indeed a judgment (entailment or contradiction) on the arguments' pairs, that are used as input to build the argumentation framework, as described in the next Section.

3 Second step: argumentation theory

Starting from a set of arguments and the attacks (i.e., conflicts) among them, a (Dung, 1995)-style argumentation framework allows to detect which are the accepted arguments. Such arguments are considered as believable by an external evaluator who has a full knowledge of the argumentation framework, and they are determined through the acceptability semantics (Dung, 1995). Roughly, an argument is accepted, if all the arguments attacking it are rejected, and it is rejected if it has at least an argument attacking it which is accepted. An argument which is not attacked at all is accepted.

Definition 1. *An abstract argumentation framework (AF) is a pair $\langle \mathcal{A}, \rightarrow \rangle$ where $\mathcal{A}$ is a set of arguments and $\rightarrow \subseteq \mathcal{A} \times \mathcal{A}$ is a binary relation called attack.*

Aim of the argumentation-based reasoning step is to provide the participant with a complete view on the arguments proposed in the debate, and to show which are the accepted ones. In our framework, we first map contradiction with the attack relation in abstract argumentation; second, the entailment relation is viewed as a support relation among abstract arguments. The support relation (Cayrol and Lagasquie-Schiex, 2011) may be represented as: (1) a relation among the arguments which does not affect their acceptability, or (2) a relation among the arguments which leads to the introduction of additional attacks.

Consider a support relation among two arguments, namely A_i and A_j. If we choose (1), an attack towards A_i or A_j does not affect the acceptability of A_j or A_i, respectively. If we choose (2), we introduce additional attacks, and we have the following two options: [Type 1] A_i *supports* A_j then A_k *attacks* A_j, and [Type 2] A_i *supports* A_j then A_k *attacks* A_i. The attacks of type 1 are due to inference: A_i entails A_j means that A_i is more specific of A_j, thus an attack towards A_j is an attack also towards A_i. The attacks of type 2, instead, are more rare, but they may happen in debates: an attack towards the more specific argument A_i is an attack towards the more general argument A_j. In Section 4, we will consider only the introduction of attacks of type 1.

For Examples 1, 2, and 3, the TE phase returns the following couples: T1 *entails* H, T2 *attacks* H, T3 *attacks* H1 (i.e. T2). The argumentation module

maps each element to its corresponding argument: H $\equiv A_1$, T1 $\equiv A_2$, T2 $\equiv A_3$, and T3 $\equiv A_4$. The resulting AF (Figure 1) shows that the accepted arguments are $\{A_1, A_2, A_4\}$, meaning that the issue "The use of cell-phones while driving is a public hazard" (A_1) is considered as accepted. Figure 2 visualizes the complete framework of the debate "Use of cell phones while driving" on Debatepedia. Accepted arguments are double bordered.

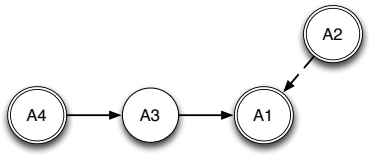


Figure 1: The AF built from the results of the TE module for Example 1, 2 and 3, without introducing additional attacks. Plain arrows represent *attacks*, dashed arrows represent *supports*.

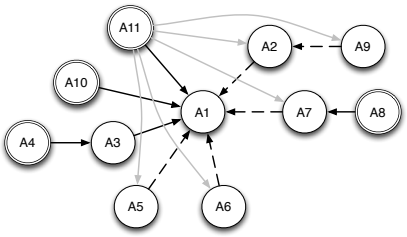


Figure 2: The AF built from the results of the TE module for the entire debate. Grey attacks are of type 1. For picture clarity, we introduce type 1 attacks only from A_{11}. The same attacks hold from A_{10} and A_3.

4 Experimental setting

We experiment the combination of TE and argumentation theory to support the interaction of online debates participants on Debatepedia, an encyclopedia of pro and con arguments on critical issues.

Data set. To create the data set of arguments pairs to evaluate our task[3], we randomly selected a set of topics (reported in column *Topics*, Table 1) of Debatepedia debates, and for each topic we coupled all the pros and cons arguments both with the main argument (the issue of the debate, as in Example 1 and 2) and/or with other arguments to which the most recent argument refers, e.g., Example 3. Using Debatepedia as case study provides us with already annotated arguments (*pro* ⇒ *entailment*[4], and *cons* ⇒ *contradiction*), and casts our task as a yes/no entailment task. As shown in Table 1, we collected 200 T-H pairs, 100 used to train the TE system, and 100 to test it (each data set is composed by 55 entailment and 45 contradiction pairs).[5] Test set pairs concern completely new topics, never seen by the system.

TE system. To detect which kind of relation underlies each couple of arguments, we used the EDITS system (Edit Distance Textual Entailment Suite), an open-source software package for recognizing TE[6] (Kouylekov and Negri, 2010). EDITS implements a distance-based framework which assumes that the probability of an entailment relation between a given T-H pair is inversely proportional to the distance between T and H. Within this framework, the system implements different approaches to distance computation, providing both edit distance algorithms and similarity algorithms.

Evaluation. To evaluate our combined approach, we carry out a two-step evaluation: we assess *(i)* the performances of the TE system to correctly assign the entailment/contradiction relations to the pairs of arguments in the Debatepedia data set; *(ii)* how much such performances impact on the goals of the argumentation module, i.e. how much a wrong assignment of a relation between two arguments leads to an incorrect evaluation of the accepted arguments.

For the first evaluation, we run the EDITS system off-the-shelf on the Debatepedia data set, applying one of its basic configurations (i.e. the distance entailment engine combines cosine similarity as the core distance algorithm; distance calculated on lemmas; stopword list included). EDITS accuracy on the training set is 0.69, on the test set 0.67 (a baseline applying a Word Overlap algorithm on tokenized text is also considered, and obtains an accuracy of 0.61 on the training set and 0.62 on the test set). Even using a basic configuration of EDITS, and a small data set (100 pairs for training) performances

[3]Data available for the RTE challenges are not suitable for our goal, since the pairs are extracted from news and are not linked among each other (they do not report opinions on a certain topic). `http://www.nist.gov/tac/2010/RTE/`

[4]Arguments "supporting" another argument without inference are left for future work.

[5]Available at `http://bit.ly/debatepedia_ds`

[6]Version 3.0 available at `http://edits.fbk.eu/`

Training set					Test set				
Topic	**#argum**	**#pairs**			**Topic**	**#argum**	**#pairs**		
		TOT.	**yes**	**no**			**TOT.**	**yes**	**no**
Violent games boost aggressiveness	16	15	8	7	*Ground zero mosque*	9	8	3	5
China one-child policy	11	10	6	4	*Mandatory military service*	11	10	3	7
Consider coca as a narcotic	15	14	7	7	*No fly zone over Libya*	11	10	6	4
Child beauty contests	12	11	7	4	*Airport security profiling*	9	8	4	4
Arming Libyan rebels	10	9	4	5	*Solar energy*	16	15	11	4
Random alcohol breath tests	8	7	4	3	*Natural gas vehicles*	12	11	5	6
Osama death photo	11	10	5	5	*Use of cell phones while driving*	11	10	5	5
Privatizing social security	11	10	5	5	*Marijuana legalization*	17	16	10	6
Internet access as a right	15	14	9	5	*Gay marriage as a right*	7	6	4	2
					Vegetarianism	7	6	4	2
TOTAL	109	**100**	**55**	**45**	**TOTAL**	110	**100**	**55**	**45**

Table 1: The Debatepedia data set.

on Debatepedia test set are promising, and in line with performances of TE systems on RTE data sets.

As a second step of the evaluation, we consider the impact of EDITS performances on arguments acceptability, i.e., how much a wrong assignment of a relation to a pair of arguments affects the computation of the set of accepted arguments. We identify the accepted arguments both in the correct AF of each Debatepedia debate of the data set (the gold-standard, where relations are correctly assigned), and on the AF generated basing on the relations assigned by EDITS. Our combined approach obtained the following performances: precision 0.74, recall 0.76, accuracy 0.75, meaning that the TE system mistakes in relation assignment propagate in the AF, but results are still satisfying and foster further research in this direction.

5 Related work

DebateGraph[7] is an online system for debates, but it is not grounded on argument theory to decide the accepted arguments. Chasnevar and Maguitman's (2004) system provides recommendations on language patterns using indices computed from Web corpora and defeasible argumentation. No NLP is used for automatic arguments detection. Carenini and Moore (2006) present a computational framework to generate evaluative arguments. Based on users' preferences, arguments are produced following argumentation guidelines to structure evaluative arguments. Then, NL Generation techniques are applied to return the argument in natural language. Unlike them, we do not create the arguments, but we use TE to detect them in texts, and we use Dung's model to identify the accepted ones. Wyner and van Engers (2010) present a policy making support tool based on forums, where NLP and argumentation are coupled to provide well structured statements. Beside the goal, several points distinguish our proposal from this one: *(i)* the user is asked to write the input text using Attempt to Controlled English, with a restricted grammar and vocabulary, while we do not support the participant in writing the text, but we automatically detect the arguments (no language restriction); *(ii)* a mode indicates the relations between the statements, while we infer them using TE; *(iii)* no evaluation of their framework is provided.

6 Future challenges

Several research lines are considered to improve the proposed framework: first, the use of NLP to detect the arguments from text will make argumentation theory applicable to reason in real scenarios. We plan to use the TE module to reason on the introduction of the support relation in abstract argumentation theory. We plan to extend our model by considering also other kinds of relationships among the arguments. Moreover, given the promising results we obtained, we plan to extend the experimentation setting both increasing the size of the Debatepedia data set, and to improve the TE system performances to apply our combined approach in other real applications (considering for instance the presence of unrelated arguments, e.g. texts that do not entail nor contradict).

[7] http://debategraph.org

References

Barzilay R. and McKeown K.R. 2005. *Sentence fusion for multidocument news summarization*. Computational Linguistics, 31(3). pp. 297-327.

Carenini G. and Moore J.D. 2006. *Generating and evaluating evaluative arguments*. Artificial Intelligence, volume 170, n. 11. pp. 925-952.

Cayrol C. and Lagasquie-Schiex M.C. 2011. *Bipolarity in Argumentation Graphs: Towards a Better Understanding*. Proceedings of SUM 2011. pp.137-148

Chesñevar C.I. and Maguitman A.G. 2004. *An Argumentative Approach to Assessing Natural Language Usage based on the Web Corpus*. Proceedings of ECAI. pp.581-585.

Dagan I. and Dolan B. and Magnini B. and Roth D. 2009. *Recognizing textual entailment: Rational, evaluation and approaches*. Natural Language Engineering (JNLE), Special Issue 04, volume 15. pp. i-xvii. Cambridge University Press.

Dung P.M. 1995. *On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games*. Artificial Intelligence, volume 77, n.2. pp.321-358.

Kouylekov M. and Negri M. 2010. *An Open-Source Package for Recognizing Textual Entailment*. Proceedings of ACL 2010 System Demonstrations. pp.42-47.

Nielsen R.D. and Ward W. and Martin J.H. 2009. *Recognizing entailment in intelligent tutoring systems*. The Journal of Natural Language Engineering, (JNLE), volume 15. pp. 479-501. Cambridge University Press.

Romano L. and Kouylekov M. O. and Szpektor I. and Dagan I. and Lavelli A. 2006. *Investigating a Generic Paraphrase-Based Approach for Relation Extraction*. Proceedings of EACL 2006. pp. 409-416.

Wyner A. and van Engers T. 2010. *A framework for enriched, controlled on-line discussion forums for e-government policy-making*. Proceedings of eGov 2010.

Towards the Unsupervised Acquisition of Discourse Relations

Christian Chiarcos
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292
chiarcos@daad-alumni.de

Abstract

This paper describes a novel approach towards the empirical approximation of discourse relations between different utterances in texts. Following the idea that every pair of events comes with preferences regarding the range and frequency of discourse relations connecting both parts, the paper investigates whether these preferences are manifested in the distribution of relation words (that serve to signal these relations).

Experiments on two large-scale English web corpora show that significant correlations between pairs of adjacent events and relation words exist, that they are reproducible on different data sets, and for three relation words, that their distribution corresponds to theory-based assumptions.

1 Motivation

Texts are not merely accumulations of isolated utterances, but the arrangement of utterances conveys *meaning*; human text understanding can thus be described as a process to recover the global structure of texts and the relations linking its different parts (Vallduví 1992; Gernsbacher et al. 2004). To capture these aspects of meaning in NLP, it is necessary to develop operationalizable theories, and, within a supervised approach, large amounts of annotated training data. To facilitate manual annotation, weakly supervised or unsupervised techniques can be applied as preprocessing step for *semi*manual annotation, and this is part of the motivation of the approach described here.

Discourse relations involve different aspects of meaning. This may include factual knowledge about the connected discourse segments (a 'subject-matter' relation, e.g., if one utterance represents the cause for another, Mann and Thompson 1988, p.257), argumentative purposes (a 'presentational' relation, e.g., one utterance motivates the reader to accept a claim formulated in another utterance, ibid., p.257), or relations between entities mentioned in the connected discourse segments (anaphoric relations, Webber et al. 2003). Discourse relations can be indicated explicitly by optional cues, e.g., adverbials (e.g., *however*), conjunctions (e.g., *but*), or complex phrases (e.g., *in contrast to what Peter said a minute ago*). Here, these cues are referred to as *relation words*.

Assuming that relation words are associated with specific discourse relations (Knott and Dale 1994; Prasad et al. 2008), the distribution of relation words found between two (types of) events can yield insights into the range of discourse relations possible at this occasion and their respective likeliness. For this purpose, this paper proposes a background knowledge base (BKB) that hosts pairs of events (here heuristically represented by verbs) along with distributional profiles for relation words. The primary data structure of the BKB is a *triple* where one event (type) is connected with a particular relation word to another event (type). Triples are further augmented with a *frequency score* (expressing the likelihood of the triple to be observed), a *significance score* (see below), and a *correlation score* (indicating whether a pair of events has a positive or negative correlation with a particular relation word).

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 213–217,
Jeju, Republic of Korea, 8-14 July 2012.

Triples can be easily acquired from automatically parsed corpora. While the relation word is usually part of the utterance that represents the source of the relation, determining the appropriate target (antecedent) of the relation may be difficult to achieve. As a heuristic, an adjacency preference is adopted, i.e., the target is identified with the main event of the preceding utterance.[1] The BKB can be constructed from a sufficiently large corpus as follows:

- identify event types and relation words
- for every utterance
  - create a candidate triple consisting of the event type of the utterance, the relation word, and the event type of the preceding utterance.
  - add the candidate triple to the BKB, if it found in the BKB, increase its score by (or initialize it with) 1,
- perform a pruning on all candidate triples, calculate significance and correlation scores

Pruning uses statistical significance tests to evaluate whether the relative frequency of a relation word for a pair of events is significantly higher or lower than the relative frequency of the relation word in the entire corpus. Assuming that incorrect candidate triples (i.e., where the factual target of the relation was non-adjacent) are equally distributed, they should be filtered out by the significance tests.

The goal of this paper is to evaluate the validity of this approach.

2 Experimental Setup

By generalizing over multiple occurrences of the same events (or, more precisely, event types), one can identify preferences of event pairs for one or several relation words. These preferences capture *context-invariant* characteristics of pairs of events and are thus to considered to reflect a semantic predisposition for a particular discourse relation.

Formally, an event is the semantic representation of the meaning conveyed in the utterance. We assume that the same event can reoccur in different contexts, we are thus studying relations between *types* of events. For the experiment described here, events are heuristically identified with the main predicates of a sentence, i.e., non-auxiliar, non-causative, non-modal verbal lexemes that serve as heads of main clauses.

The primary data structure of the approach described here is a triple consisting of a source event, a relation word and a target (antecedent) event. These triples are harvested from large syntactically annotated corpora. For intersentential relations, the target is identified with the event of the immediately preceding main clause. These extraction preferences are heuristic approximations, and thus, an additional pruning step is necessary.

For this purpose, statistical significance tests are adopted (χ^2 for triples of frequent events and relation words, t-test for rare events and/or relation words) that compare the relative frequency of a relation word given a pair of events with the relative frequency of the relation word in the entire corpus. All results with $p \geq .05$ are excluded, i.e., only triples are preserved for which the observed positive or negative correlation between a pair of events and a relation word is not due to chance with at least 95% probability. Assuming an even distribution of incorrect target events, this should rule these out. Additionally, it also serves as a means of evaluation. Using statistical significance tests as pruning criterion entails that all triples eventually confirmed are statistically significant.[2]

This setup requires *immense* amounts of data: We are dealing with several thousand events (theoretically, the total number of verbs of a language). The chance probability for two events to occur in adjacent position is thus far below 10^{-6}, and it decreases further if the likelihood of a relation word is taken into consideration. All things being equal, we thus need *millions* of sentences to create the BKB.

Here, two large-scale corpora of English are employed, PukWaC and Wackypedia_EN (Baroni et al. 2009). PukWaC is a 2G-token web corpus of British English crawled from the `uk` domain (Ferraresi et al.

[1] Relations between non-adjacent utterances are constrained by the structure of discourse (Webber 1991), and thus less likely than relations between adjacent utterances.

[2] Subsequent studies may employ less rigid pruning criteria. For the purpose of the current paper, however, the statistical significance of all extracted triples serves as an criterion to evaluate methodological validity.

2008), and parsed with MaltParser (Nivre et al. 2006). It is distributed in 5 parts; Only PukWaC-1 to PukWaC-4 were considered here, constituting 82.2% (72.5M sentences) of the entire corpus, PukWaC-5 is left untouched for forthcoming evaluation experiments. Wackypedia_EN is a 0.8G-token dump of the English Wikipedia, annotated with the same tools. It is distributed in 4 different files; the last portion was left untouched for forthcoming evaluation experiments. The portion analyzed here comprises 33.2M sentences, 75.9% of the corpus.

The extraction of events in these corpora uses simple patterns that combine dependency information and part-of-speech tags to retrieve the main verbs and store their lemmata as event types. The target (antecedent) event was identified with the last main event of the preceding sentence. As relation words, only sentence-initial children of the source event that were annotated as adverbial modifiers, verb modifiers or conjunctions were considered.

PukWaC (sub)corpus		Wackypedia_EN triples		
sentences	triples	common	agreeing	%
1.2M	74	20	12	60.0
4.8M	832	177	132	75.5
19.2M	7,342	938	809	86.3
38.4M	20,106	1,783	1,596	89.9
72.5M	46,680	2,643	2,393	90.5

Table 1: Agreement with respect to positive or negative correlation of event pairs and relation words between Wackypedia_EN and PukWaC subcorpora of different size

	PukWaC triples			agreement (%)	
	total	vs. H	vs. T	vs. H	vs. T
B: *but*	11,042	6,805	1,525	97.7	62.2
H: *however*	7,251		1,413		66.9
T: *then*	1,791				

Table 2: Agreement between *but* (B), *however* (H) and *then* (T) on PukWaC

3 Evaluation

To evaluate the validity of the approach, three fundamental questions need to be addressed: **significance** (are there significant correlations between pairs of events and relation words ?), **reproducibility** (can these correlations confirmed on independent data sets ?), and **interpretability** (can these correlations be interpreted in terms of theoretically-defined discourse relations ?).

3.1 Significance and Reproducibility

Significance tests are part of the pruning stage of the algorithm. Therefore, the number of triples eventually retrieved confirms the existence of statistically significant correlations between pairs of events and relation words. The left column of Tab. 1 shows the number of triples obtained from PukWaC subcorpora of different size.

For reproducibility, compare the triples identified with Wackypedia_EN and PukWaC subcorpora of different size: Table 1 shows the number of triples found in both Wackypedia_EN and PukWaC, and the *agreement* between both resources. For two triples involving the same events (event types) and the same relation word, agreement means that the relation word shows either positive or negative correlation in both corpora, disagreement means positive correlation in one corpus and negative correlation in the other.

Table 1 confirms that results obtained on one resource can be reproduced on another. This indicates that triples indeed capture context-invariant, and hence, semantic, characteristics of the relation between events. The data also indicates that reproducibility increases with the size of corpora from which a BKB is built.

3.2 Interpretability

Any theory of discourse relations would predict that relation words with similar function should have similar distributions, whereas one would expect different distributions for functionally unrelated relation words. These expectations are tested here for three of the most frequent relation words found in the corpora, i.e., *but*, *then* and *however*. *But* and *however* can be grouped together under a generalized notion of contrast (Knott and Dale 1994; Prasad et al. 2008); *then*, on the other hand, indicates a temporal and/or causal relation.

Table 2 confirms the expectation that event pairs that are correlated with *but* tend to show the same correlation with *however*, but not with *then*.

4 Discussion and Outlook

This paper described a novel approach towards the unsupervised acquisition of discourse relations, with encouraging preliminary results: Large collections of parsed text are used to assess distributional profiles of relation words that indicate discourse relations that are possible between specific types of events; on this basis, a background knowledge base (BKB) was created that can be used to predict an appropriate discourse marker to connect two utterances with no overt relation word.

This information can be used, for example, to facilitate the semiautomated annotation of discourse relations, by pointing out the 'default' relation word for a given pair of events. Similarly, Zhou et al. (2010) used a language model to predict discourse markers for implicitly realized discourse relations. As opposed to this shallow, n-gram-based approach, here, the internal structure of utterances is exploited: based on semantic considerations, syntactic patterns have been devised that extract triples of event pairs and relation words. The resulting BKB provides a distributional approximation of the discourse relations that can hold between two specific event types. Both approaches exploit complementary sources of knowledge, and may be combined with each other to achieve a more precise prediction of implicit discourse connectives.

The validity of the approach was evaluated with respect to three evaluation criteria: The extracted associations between relation words and event pairs could be shown to be statistically significant, and to be reproducible on other corpora; for three highly frequent relation words, theoretical predictions about their relative distribution could be confirmed, indicating their interpretability in terms of presupposed taxonomies of discourse relations.

Another prospective field of application can be seen in NLP applications, where selection preferences for relation words may serve as a cheap replacement for full-fledged discourse parsing. In the Natural Language Understanding domain, the BKB may help to disambiguate or to identify discourse relations between different events; in the context of Machine Translation, it may represent a factor guiding the insertion of relation words, a task that has been found to be problematic for languages that differ in their inventory and usage of discourse markers, e.g., German and English (Stede and Schmitz 2000). The approach is language-independent (except for the syntactic extraction patterns), and it does not require manually annotated data. It would thus be easy to create background knowledge bases with relation words for other languages or specific domains – given a sufficient amount of textual data.

Related research includes, for example, the unsupervised recognition of causal and temporal relationships, as required, for example, for the recognition of textual entailment. Riaz and Girju (2010) exploit distributional information about pairs of utterances. Unlike approach described here, they are not restricted to adjacent utterances, and do not rely on explicit and recurrent relation words. Their approach can thus be applied to comparably small data sets. However, they are restricted to a specific type of relations whereas here the entire bandwidth of discourse relations that are explicitly realized in a language are covered. Prospectively, both approaches could be combined to compensate their respective weaknesses.

Similar observations can be made with respect to Chambers and Jurafsky (2009) and Kasch and Oates (2010), who also study a single discourse relation (narration), and are thus more limited in scope than the approach described here. However, as their approach extends beyond pairs of events to complex event chains, it seems that both approaches provide complementary types of information and their results could also be combined in a fruitful way to achieve a more detailed assessment of discourse relations.

The goal of this paper was to evaluate the methdological validity of the approach. It thus represents the basis for further experiments, e.g., with respect to the enrichment the BKB with information provided by Riaz and Girju (2010), Chambers and Jurafsky (2009) and Kasch and Oates (2010). Other directions of subsequent research may include address more elaborate models of events, and the investigation of the relationship between relation words and taxonomies of discourse relations.

Acknowledgments

This work was supported by a fellowship within the Postdoc program of the German Academic Exchange Service (DAAD). Initial experiments were conducted at the Collaborative Research Center (SFB) 632 "Information Structure" at the University of Potsdam, Germany. I would also like to thank three anonymous reviewers for valuable comments and feedback, as well as Manfred Stede and Ed Hovy whose work on discourse relations on the one hand and proposition stores on the other hand have been the main inspiration for this paper.

References

M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009.

N. Chambers and D. Jurafsky. Unsupervised learning of narrative schemas and their participants. 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*, pages 602–610. Association for Computational Linguistics, 2009.

A. Ferraresi, E. Zanchetta, M. Baroni, and S. Bernardini. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54, 2008.

Morton Ann Gernsbacher, Rachel R. W. Robertson, Paola Palladino, and Necia K. Werner. Managing mental representations during narrative comprehension. *Discourse Processes*, 37(2):145–164, 2004.

N. Kasch and T. Oates. Mining script-like structures from the web. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, pages 34–42. Association for Computational Linguistics, 2010.

A. Knott and R. Dale. Using linguistic phenomena to motivate a set of coherence relations. *Discourse processes*, 18(1):35–62, 1994.

J. van Kuppevelt and R. Smith, editors. *Current Directions in Discourse and Dialogue*. Kluwer, Dordrecht, 2003.

William C. Mann and Sandra A. Thompson. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

J. Nivre, J. Hall, and J. Nilsson. Maltparser: A data-driven parser-generator for dependency parsing. In *Proc. of LREC*, pages 2216–2219. Citeseer, 2006.

R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. Joshi, and B. Webber. The penn discourse treebank 2.0. In *Proc. 6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, 2008.

M. Riaz and R. Girju. Another look at causality: Discovering scenario-specific contingency relationships with no supervision. In *Semantic Computing (ICSC), 2010 IEEE Fourth International Conference on*, pages 361–368. IEEE, 2010.

M. Stede and B. Schmitz. Discourse particles and discourse functions. *Machine translation*, 15(1): 125–147, 2000.

Enric Vallduví. *The Informational Component*. Garland, New York, 1992.

Bonnie L. Webber. Structure and ostension in the interpretation of discourse deixis. *Natural Language and Cognitive Processes*, 2(6):107–135, 1991.

Bonnie L. Webber, Matthew Stone, Aravind K. Joshi, and Alistair Knott. Anaphora and discourse structure. *Computational Linguistics*, 4(29):545–587, 2003.

Z.-M. Zhou, Y. Xu, Z.-Y. Niu, M. Lan, J. Su, and C.L. Tan. Predicting discourse connectives for implicit discourse relation recognition. In *COLING 2010*, pages 1507–1514, Beijing, China, August 2010.

Arabic Retrieval Revisited: Morphological Hole Filling

Kareem Darwish, Ahmed M. Ali
Qatar Computing Research Institute
Qatar Foundation, Doha, Qatar
kdarwish@qf.org.qa, amali@qf.org.qa

Abstract

Due to Arabic's morphological complexity, Arabic retrieval benefits greatly from morphological analysis – particularly stemming. However, the best known stemming does not handle linguistic phenomena such as broken plurals and malformed stems. In this paper we propose a model of character-level morphological transformation that is trained using Wikipedia hypertext to page title links. The use of our model yields statistically significant improvements in Arabic retrieval over the use of the best statistical stemming technique. The technique can potentially be applied to other languages.

1. Introduction

Arabic exhibits rich morphological phenomena that complicate retrieval. Arabic nouns and verbs are typically derived from a set of 10,000 roots that are cast into stems using templates that may add infixes, double letters, or remove letters. Stems can accept the attachment of clitics, in the form of prefixes or suffixes, such as prepositions, determiners, pronouns, etc. Orthographic rules can cause the addition, deletion, or substitution of letters during suffix and prefix attachment. Further, stems can be inflected to obtain plural forms via the addition of suffixes or through using a different stem form altogether producing so-called broken[1] (aka irregular) plurals.

For retrieval, we would ideally like to match "related" stem forms regardless of inflected form or attached clitic. Tolerating some form of derivational morphology where nouns are transformed into adjectives via the attachment of the suffix ي (y)[2] (ex. مصر (mSr) ➔ مصري (mSry)) is desirable as they are semantically related. Matching all stems that are cast from the same root would introduce undesired ambiguity, because a single root can produce up to 1,000 stems.

Two general approaches have been shown to improve Arabic retrieval. The first approach involves stemming, which removes clitics, plural and gender markers, and suffixes such as ي (y). Statistical stemming was reported to be the most effective for Arabic retrieval (Darwish et al., 2005). Though effective, stemming has the following drawbacks:

1. Stemming does not handle infixes and hence cannot conflate singular and broken plural word forms. For example, the plural of the Arabic word for book "كتاب" (ktAb) is "كتب" (ktb).
2. Stemming of some named entities, which are important for retrieval, and their inflected forms may produce different stems as word endings may change with the attachment of suffixes. Consider the Arabic words for America أمريكا (>mrykA) and American أمريكي (>mryky), where the final letter is transformed from "A" to "y".

The second approach involves using character 3- or 4-grams (as opposed to words) (Mayfield et al., 2001; Darwish and Oard, 2002). For example, the trigrams of "WORD" are "WOR" and "ORD". This approach though it has been shown to improve retrieval effectiveness, it has the following drawbacks:

1. It cannot handle broken plurals, though it would handle words where stemming would produce different stems for different inflected forms.
2. It significantly increases index sizes. For example, using a 6 letter word would produce 4 trigram chunks, which would have 12 letters.
3. Longer words would yield more character n-gram chunks compared to shorter ones leading to skewed weights for query words.

[1] "Broken" is a direct translation of the Arabic word "takseer", which refers to this kind of plural.

[2] We use Buckwalter transliteration in the paper

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 218–222,
Jeju, Republic of Korea, 8-14 July 2012.

To address this problem, we propose the use of a character level transformation model that can generate tokens that are morphologically related to query tokens. We train the model using morphological related stems that are extracted from hypertext/page title pairs from Wikipedia. Such pairs are good for the task at hand, because they show different ways to refer to the same concept. We show that expanding stems in a query with related stems using our model outperforms the use of state-of-the-art statistical Arabic stemming. Further, the expansion can be applied to words directly to perform at par with statistical stemming. Laterally, the model can help produce spelling variants of transliterated names.

The contribution of this paper is as follows:

- We proposed an automatic method for learning character-level morphological transformations from Wikipedia hypertext/page title pairs.
- When applied to stems, we show that the method overcomes some morphological problems that are associated with stemming, statistically significantly outperforming Arabic retrieval using statistical stemming and character n-grams.
- When applied to words, we show that the method yields retrieval effectiveness at par with statistical stemming.

2. Related Work

Most studies are based on a single large collection from the TREC-2001/2002 cross-language retrieval track (Gey and Oard, 2001; Oard and Gey, 2002). The studies examined indexing using words, word clusters (Larkey et al., 2002), terms obtained through morphological analysis (e.g., stems and roots (Darwish and Oard, 2002), light stemming (Aljlayl et al., 2001; Larkey et al., 2002), and character n-grams of various lengths (Darwish and Oard, 2002; Mayfield et al., 2001). The effects of normalizing alternative characters, removal of diacritics and stop-word removal have also been explored (Xu et al., 2001). These studies suggest that light stemming, character n-grams, and statistical stemming are the better index terms. Morphological approaches assume an Arabic word is constituted from prefixes-stem-suffixes and aim to remove prefixes and suffixes. Since Arabic morphology is ambiguous, statistical stemming attempts to find the most likely segmentation of words. The first such systems were MORPHO3 (Ahmed, 2000) and Sebawai (Darwish, 2002). Later work by Lee et al. (2003) used a trigram language model with a minimal set of manually crafted rules to achieve a stemming accuracy of 97.1%. Their system was shown by Darwish et al. (2005) to lead to statistical improvements over using light stemming. Diab (2009) used an SVM classifier to ascertain the optimal segmentation for a word in context. The classifier was trained on the Arabic Penn Treebank data. She reported a stemming accuracy of 99.2%. Although consistency is more important for IR applications than linguistic correctness, perhaps improved correctness would naturally yield great consistency. In this paper, we used a reimplementation of the system proposed by Diab (2009) with the same training set as a baseline.

Concerning the automatic induction of morphologically related word-forms, Hammarström (2009) surveyed fairly comprehensively many unsupervised morphology learning approaches. Brent et al. (1995) proposed the use of Minimum Description Length (MDL) to automatically discover suffixes. MDL based approach was improved by: Goldsmith (2001) who applied the EM algorithm to improve the precision of pairing stems prior to suffix induction; and Schone and Jurafsky (2001) who applied latent semantic analysis to determine if two words are semantically related. Jacquemin (1997) used word grams that look similar, i.e. share common stems, to learn suffixes. Baroni (2002) extended his work by incorporating semantic similarity features, via mutual information, and orthographic features, via edit distance. Chen and Gey (2002) utilized a bilingual dictionary to find Arabic words with a common stem that map to the same English stem. Also in the cross-language spirit, Snyder and Barzilay (2008) used cross-language mappings to learn morpheme patterns and consequently automatically segment words. They successfully applied their method to Arabic, Hebrew, and Aramaic. Creutz and Lagus (2007) proposed a probabilistic model for automatic word segment discovery. Most of these approaches can discover suffixes and prefixes without human intervention. However, they may not be able to handle infixation and spelling variations. Karagol-Ayan et al. (2006) used approximate string matching to automatically

map morphologically similar words in noisy dictionary data. They used the mappings to learn affixation, including infixation, from noisy data. In this paper, we propose a new technique for finding morphologically related word-forms based on learning character-level mappings.

Figure 1. Example hypertexts to Wikipedia titles

3. Character-Level Model

3.1 Training Data

In our experiments, we extracted Wikipedia hypertext to page title pairs as in Figure 1. We performed all work on an Arabic Wikipedia dump from April 2010, which contained roughly 150,000 articles. In all, we extracted 11.47 million hypertext-title pairs. From them, we attempted to find word pairs that were morphologically related. From the example in Figure 1, given the hypertext بالبرتغالية (bAlbrtgAlyp – in Portuguese) and the page title that it points to لغة برتغالية (lgp brtgAlyp – Portuguese language) we needed to extract the pairs بالبرتغالية (bAlbrtgAlyp) and برتغالية (brtgAlyp).

We assumed that a word in the hypertext and another in Wikipedia title were morphologically related using the following criteria:

- The words share the first 2 letters or the last 2 letters. This was intended to increase precision.
- The edit distance between the two words must be <= 3. The choice of 3 was motivated by the fact that Arabic prefixes and suffixes are typically 1, 2, or 3 letters long.
- The edit distance was less than 50% of the length of the shorter of the two words. This was important to insure that short words that share common letters but are in fact different are filtered out.

The word pairs that matched these criteria were roughly 13 million word pairs[3]. All words in the word pairs were stemmed using a reimplementation of the stemmer of Diab (2009).

3.2 Alignment and Generation

Alignment: We performed two alignments. In the first, we aligned the stems of the word pairs at character level. In the second, we aligned the words of the word pairs at character level without stemming. The pairs were aligned using Giza++ and the phrase extractor and scorer from the Moses ma-chine translation package (Koehn et al., 2007). To apply a machine translation analogy, we treated words as sentences and the letters from which were constructed as tokens. The alignment produced letter sequence mappings. Source character sequence lengths were restricted to 3 letters.

Generating related stems/words: We treated the problem of generating morphologically related stems (or words) like a transliteration mining problem akin to that in Udupa et al. (2009). Briefly, the miner used character segment mappings to generate all possible transformations while constraining generation to the existing tokens (either stems or words) in a list of unique tokens in the retrieval test collection.

Basically, given a query token, all possible segmentations, where each segment has a maximum length of 3 characters, were produced along with their associated mappings. Given all mapping combinations, combinations producing valid target tokens were retained and sorted according to the product of their mapping probabilities. To illustrate how this works, consider the following example: Given a query word "min", target words in the word list {moon, men, man, min}, and the possible mappings for the segments and their probabilities:

m = {(m, 0.7), (me, 0.25), (ma, 0.05)}
mi = {(mi, 0.5), (me, 0.3), (m, 0.15), (ma, 0.05)}
n = {n, 0.7), (nu, 0.2), (an, 0.1)}
in = {(in, 0.8), (en, 0.2)}

The algorithm would produce the following candidates with the corresponding channel probabilities:

(min➔min:0.56): (m➔m: 0.7); (in➔in: 0.8)
(min➔men:0.18): (m➔m: 0.7); (in➔en: 0.2)

[3] The training data can be obtained from: https://github.com/kdarwish/WikiPairs

(min➔man:0.035): (mi➔ma: 0.05); (n➔n: 0.7)
The implementation details of the decoder are described in (El-Kahki et al., 2012).

4. Testing Arabic Retrieval Effectiveness

4.1 Experimental Setup

We used extrinsic IR evaluation to determine the quality of the related stems that were generated. We performed experiments on the TREC 2001/2002 cross language track collection, which contains 383,872 Arabic newswire articles and 75 topics with their relevance judgments (Oard and Gey, 2002). This is presently the best available large Arabic information retrieval test collection. We used Mean Average Precision (MAP) as the measure of goodness for this retrieval task. Going down from the top a retrieved ranked list, Average Precision (AP) is the average of precision values computed at every relevant document found. MAP is just the mean of the AP's for all queries.
All experiments were performed using the Indri retrieval toolkit, which uses a retrieval model that combines inference networks and language modeling and implements advanced query operators (Metzler and Croft, 2004). We used a paired 2-tailed t-test with p-value less than 0.05 to determine if a set of retrieval results was better than another.

We replaced each query tokens with all the related stems that were generated using a weighted synonym operator (Wang and Oard, 2006), where the weights correspond to the product of the mapping probabilities for each related word. With the weighted synonym operator, we did not need to threshold the generated related stems as ones with low probabilities were demoted. Probabilities were normalized by the score of the original query word. For example, given the stem صناع (SnAE) it was replaced with: #wsyn(1.000 SnAE 0.029 SnAEy 0.013 SnE 0.006 SnAEA 0.003 mSnwE).

We used three baselines to compare against, namely: using raw words, using statistical stemming (Diab, 2009), and character 4-grams. For all runs, we performed letter normalization, where we conflated: variants of "alef", "ta marbouta" and "ha", "alef maqsoura" and "ya", and the different forms of "hamza".

Table 1. Retrieval Results

Run	MAP	Statistically better than
Words	0.225	
Stems	0.276	words
Char 4-grams	0.244	
Expanded Words	0.264	words
Expanded Stems	0.296	words/stems/char 4-grams

4.2 Experimental Results

Table 1 reports retrieval results. Expanding stems using morphologically related stems yielded statistically significant improvements over using words, stems, and character 4-grams. Expanding words yielded results that were statistically significantly better than using words, and statistically indistinguishable from using 4-grams and stems. As the results show, the proposed technique improves upon statistical stemming by overcoming the shortfalls of stemming. Another phenomenon that was addressed implicitly by the proposed technique had to do with detecting variant spellings of transliterated names. This draws from the fact that differences in spelling variations and the construction of broken plurals are typically due to the insertion or deletion of long vowels. For example, given the name "نتنياهو" (ntnyAhw– Netanyahu), the model proposed: ntynyAhw, ntAnyAhw, and ntAnyhw.

5. Conclusion

In this paper, we presented a method for generating morphologically related tokens from Wikipedia hypertext to page title pairs. We showed that the method overcomes some of the problems of statistical stemming to yield statistically significant improvements in Arabic retrieval over using statistical stemming. The technique can also be applied on words to yield results that statistically indistinguishable from statistical stemming. The technique had the added advantage of detecting variable spellings of transliterated named entities.

For future work, we would like to try the proposed technique on other languages, because it would likely be effective in automatically learning character-level morphological transformations as well as overcoming some of the problems associated with stemming. It is worthwhile to devise models that concurrently generate morphological and phonologically related tokens.

References

M. A. Ahmed. (2000). A Large-Scale Computational Processor of the Arabic Morphology, and Applications. A Master's Thesis, Faculty of Engineering, Cairo University, Cairo, Egypt.

M. Aljlayl, S. Beitzel, E. Jensen, A. Chowdhury, D. Holmes, M. Lee, D. Grossman, O. Frieder. IIT at TREC-10. In TREC. 2001. Gaithersburg, MD.

M. Baroni, J. Matiasek, H. Trost (2002). Unsupervised discovery of morphologically related words based on orthographic and semantic similarity. ACL-2002 Workshop on Morphological & Phonological Learning, pp. 48-57.

M. Brent, S. Murthy, A. Lundberg (1995). Discovering Morphemic Suffixes: A Case Study in Minimum Description Length Induction. 15th Annual Conference on the Cognitive Science Society, pp. 28-36.

A. Chen, F. Gey (2002). Building an Arabic Stemmer for Information Retrieval. TREC-2002.

M. Creutz, K. Lagus (2007). Unsupervised models for morpheme segmentation and morphology learning. Speech and Language Processing, Vol. 4, No 1:3, 2007.

K. Darwish. (2002). Building a Shallow Morphological Analyzer in One Day. ACL Workshop on Computational Approaches to Semitic Languages. 2002.

K. Darwish, H. Hassan, O. Emam (2005). Examining the Effect of Improved Context Sensitive Morphology on Arabic Information Retrieval. ACL Workshop on Computational Approaches to Semitic Languages, pp. 25–30, 2005.

K. Darwish, D. Oard. (2002). Term Selection for Searching Printed Arabic. SIGIR, 2002, p. 261 - 268.

M. Diab (2009). Second Generation Tools (AMIRA 2.0): Fast and Robust Tokenization, POS tagging, and Base Phrase Chunking. 2nd Int. Conf. on Arabic Language Resources and Tools, 2009.

A. El-Kahki, K. Darwish, M. Abdul-Wahab, A. Taei (2012). Transliteration Mining Using Large Training and Test Sets. NAACL-2012.

F. Gey, D. Oard (2001). The TREC-2001 Cross-Language Information Retrieval Track: Searching Arabic Using English, French or Arabic Queries. TREC, 2001. Gaithersburg, MD. p. 16-23.

J. Goldsmith (2001). Unsupervised Learning of the Morphology of a Natural Language. Journal of Computational Linguistics, Vol. 27:153-198, 2001.

H. Hammarström (2009). Unsupervised Learning of Morphology and the Languages of the World. Ph.D. Thesis, Dept. of CSE, Chalmers Univ. of Tech. and Univ. of Gothenburg.

C. Jacquemin (1997). Guessing morphology from terms and corpora. ACM SIGIR-1997, p.156-165.

B. Karagol-Ayan, D. Doermann, A. Weinberg (2006). Morphology Induction from Limited Noisy Data Using Approximate String Matching. 8th ACL SIG on Comp. Phonology at HLT-NAACL 2006, pp. 60–68.

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, E. Herbst (2007). Moses: Open Source Toolkit for Statistical Machine Translation, ACL-2007, demonstration session, Prague, Czech Republic, June 2007.

L. Larkey, L. Ballesteros, and M. Connell (2002). Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis. SIGIR 2002. pp. 275-282.

Y. Lee, K. Papineni, S. Roukos, O. Emam, H. Has-san (2003). Language Model Based Arabic Word Segmentation. ACL-2003, p. 399 - 406.

J. Mayfield, P. McNamee, C. Costello, C. Piatko, A. Banerjee. JHU/APL at TREC 2001: Experiments in Filtering and in Arabic, Video, and Web Retrieval. In TREC 2001. Gaithersburg, MD. p. 322-329.

D. Metzler, W. B. Croft (2004). Combining the Language Model and Inference Network Approaches to Retrieval. Information Processing and Management Special Issue on Bayesian Networks and Information Retrieval, 40(5), 735-750, 2004.

D. Oard, F. Gey (2002). The TREC 2002 Arabic/English CLIR Track. TREC-2002.

P. Schone, D. Jurafsky (2001). Knowledge-free induction of inflectional morphologies. ACL 2001.

B. Snyder, R. Barzilay (2008). Unsupervised Multilingual Learning for Morphological Segmentation. ACL-08: HLT, pp. 737–745, 2008.

R. Udupa, K. Saravanan, A. Bakalov, A. Bhole. 2009. "They Are Out There, If You Know Where to Look": Mining Transliterations of OOV Query Terms for Cross-Language Information Retrieval. ECIR-2009, Toulouse, France, 2009.

J. Wang, D. Oard (2006). Combining Bidirectional Translation and Synonymy for Cross-language Information Retrieval. SIGIR-2006, pp. 202-209.

J. Xu, A. Fraser, and R. Weischedel (2001). 2001 Cross-Lingual Retrieval at BBN. TREC 2001, pp. 68 - 75.

Extracting and modeling durations for habits and events from Twitter

Jennifer Williams
Department of Linguistics
Georgetown University
Washington, D.C., USA
jaw97@georgetown.edu

Graham Katz
Department of Linguistics
Georgetown University
Washington, D.C., USA
egk7@georgetown.edu

Abstract

We seek to automatically estimate typical durations for events and habits described in Twitter tweets. A corpus of more than 14 million tweets containing temporal duration information was collected. These tweets were classified as to their habituality status using a bootstrapped, decision tree. For each verb lemma, associated duration information was collected for episodic and habitual uses of the verb. Summary statistics for 483 verb lemmas and their typical habit and episode durations has been compiled and made available. This automatically generated duration information is broadly comparable to hand-annotation.

1 Introduction

Implicit information about temporal durations is crucial to any natural language processing task involving temporal understanding and reasoning. This information comes in many forms, among them knowledge about typical durations for events and knowledge about typical times at which an event occurs. We know that lunch lasts for half an hour to an hour and takes place around noon, a game of chess lasts from a few minutes to a few hours and can occur any time, and so when we interpret a text such as "After they ate lunch, they played a game of chess and then went to the zoo" we can infer that the zoo visit probably took place in the early afternoon. In this paper we focus on duration. Hand-annotation of event durations is expensive slow (Pan et al., 2011), so it is desirable to automatically determine typical durations. This paper describes a method for automatically extracting information about typical durations for events from tweets posted to the Twitter microblogging site.

Twitter is a rich resource for information about everyday events – people post their tweets to Twitter publicly in real-time as they conduct their activities throughout the day, resulting in a significant amount of mundane information about common events. For example, (1) and (2) were used to provide information about how long a *work* event can last:

(1) *Had **work for an hour and 30 mins** now going to disneyland with my cousins :)*

(2) *I play in a loud rock band, I **worked at a night club for two years**. My ears have never hurt so much @melaniemarnie @giorossi88 @CharlieHi11*

In this paper, we sought to use this kind information to determine likely durations for events and habits of a variety of verbs. This involved two steps: extracting a wide range of tweets such as (1) and (2) and classifying these as to whether they referred to specific event (as in (1)) or a general habit (as in (2)), then summarizing the duration information associated with each kind of use of a given verb.

This paper answers two investigative questions:

- How well can we automatically extract fine-grain duration information for events and habits from Twitter?
- Can we effectively distinguish episode and habit duration distributions ?

The results presented here show that Twitter can be mined for fine-grain event duration information

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 223–227,
Jeju, Republic of Korea, 8-14 July 2012.

with high precision using regular expressions. Additionally, verb uses can be effectively categorized as to their habituality, and duration information plays an important role in this categorization.

2 Prior Work

Past research on typical durations has made use of standard corpora with texts from literature excerpts, news stories, and full-length weblogs (Pan et al, 2006; 2007; 2011; Kozareva & Hovy, 2011; Gusev et al., 2011). For example, Pan et al. (2011) hand-annotated of a portion of the TIMEBANK corpus that consisted of Wall Street Journal articles. For 58 non-financial articles, they annotated over 2,200 events with typical temporal duration, specifying the upper and lower bounds for the duration of each event. In addition they used their corpus to automatically determine event durations with machine learning, predicting features of the duration on the basis of the verb lemma, local textual context. and other information. Their best (SVM) classifier achieved precision of 78.2% on the course-grained task of determining whether an event's duration was longer or shorter than one day (compared with 87.7% human agreement). For determining the fine-grained task of determining the most likely temporal unit–second, minute, hour, day, week, etc.–achieved 67.9% (human agreement: 79.8%). This shows that lexical information can be effectively leveraged for duration prediction.

To compile temporal duration information for a wider range of verbs, Gusev et al. (2011) explored an automatic Web-based query method for harvesting typical durations of events. Their data consisted of search engine "hit-counts" and they analyzed the distribution of durations associated with 1000 frequent verbs in terms of whether the event lasts for more or less than a day (course-grain task) or whether it lasts for seconds, minutes, hours, days, weeks, months, or years (fine-grain task). They note that many verbs have a two-peaked distribution and they suggest that the two-peaked distribution could be a result of the usage referring to a habit or a single episode. (When used with a duration marker, *run*, for example, is used about 15% of the time with hour-scale and 38% with year-scale duration markers). Rather than making a distinction between habits and episodes in their data, they apply a heuristic to focus on episodes only.

Kozareva and Hovy (2011) also collected typical durations of events using Web query patterns. They proposed a six-way classification of ways in which events are related to time, but provided only programmatic analyses of a few verbs using Web-based query patterns. They have proposed a compilation of the 5,000 most common verbs along with their typical temporal durations. In each of these efforts, automatically collecting a large amount of reliable to cover a wide range of verbs has been noted as a difficulty. It is this task that we seek to take up.

3 Corpus Methodology

Our goal was to discover the duration distribution as well as typical habit and typical episode durations for each verb lemma that we found in our collection. A wide range of factors influence typical event durations. Among these are the character of a verb's arguments, the presence of negation and other embedding features. For this preliminary work, we ignored the effects of arguments, and focused only on generating duration information for verb lemmas. Also, tweets that were negated, conditional tweets, and tweets in the future tense were put aside.

3.1 Data Collection

A corpus of tweets was collected from the Twitter web service API using an open-source module called Tweetstream (Halvorsen & Schierkolk, 2010). Tweets were collected that contained reference to a temporal duration. The data collection task began on February 1, 2011 and ended on September 28, 2011. Duplicate tweets were identified by their unique tweet ID provided by Twitter, and were removed from the data set. Also tweets that were marked by Twitter as 'retweets' (tweets that have been reposted to Twitter) were removed. The following query terms (denoting temporal duration measure) were used to filter the Twitter stream for tweets containing temporal duration:

> *second, seconds, minute, minutes, hour, hours, day, days, week, weeks, month, months, year, years, decade, decades, century, centuries, sec, secs, min, mins, hr, hrs, wk, wks, yr, yrs*

The number of tweets in the resulting corpus was 14,801,607 and the total number of words in the

corpus was 224,623,447. Tweets were normalized, tokenized, and then tagged for POS, using the NLTK Treebank Tagger (Bird & Loper, 2004).

3.2 Extraction Frames

To associate each temporal duration with its event, events and durations were identified and extracted using four types of regular expression extraction frames. The patterns applied a heuristic to associate each verb with a temporal expression, similar to the extraction frames used in Gusev et al. (2011). The four types of extraction frames were:

- *verb for duration*
- *verb* in *duration*
- spend *duration verb*ing
- takes *duration* to *verb*

where *verb* is the target verb and *duration* is a duration-measure term. In (3), for example, the verb *work* is associated with the temporal duration term *44 years*.

(3) *Retired watchmaker **worked for 44 years** without a telephone, to avoid unnecessary interruptions, http://t.co/ox3mB6g*

These four extraction frame types were also varied to include different tenses, different grammatical aspects, and optional verb arguments to reach a wide range of event mentions and ordering between the verb and the duration clause. For each matched tweet a feature vector was created with the following features: verb lemma, temporal bucket (seconds, minutes, hours, weeks, days, months or years), tense (past or present), grammatical aspect (simple, progressive, or perfect), duration in seconds, and the extraction frame type (for, in, spend, or take). For example, the features extracted from (3) were:

[work, years, past, simple, 1387584000, FOR]

Tweets with verbal lemmas that occur fewer than 100 times in the extracted corpus were filtered out. The resulting data set contained 390,562 feature vectors covering 483 verb lemmas.

3.3 Extraction Precision

Extraction frame performance was estimated using precision on a random sample of 400 hand-labeled tweets. Each instance in the sample was labeled as correct if the extracted feature vector was correct in its entirety. The overall precision for extraction frames was estimated as 90.25%, calculated using a two-tailed t-test for sample size of proportions with 95% confidence (p=0.05, n=400).

3.4 Duration Results

In order to summarize information about duration for each of the 483 verb lemmas, we calculated the frequency distribution of tweets by duration in seconds. This distribution can be represented in histogram form, as in Figure 1 for the verb lemma *search*, with with bins corresponding to temporal units of measure (seconds, minutes, etc.).

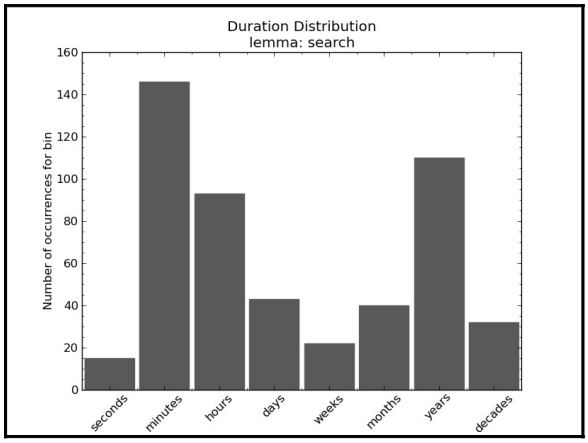


Figure 1: Frequency distribution for *search*

This histogram shows the characteristic bimodal-distributions noted by Pan et al., (2011) and Gusev et. al., (2011), an issue taken up in the next section.

4 Episodic/Habitual Classification

Most verbs have both episodic and habitual uses, which clearly correspond to different typical durations. In order to draw this distinction we built a system to automatically classify our tweets as to their habituality. The extracted feature vectors were used in a machine learning task to label each tweet in the collection as denoting a habit or an episode, broadly following Mathew & Katz (2009). This classification was done with bootstrapping, in a partially supervised manner.

4.1 Bootstrapping Classifier

First, a random sample of 1000 tweets from the extracted corpus was hand-labeled as being either

habit or episode (236 habits; 764 episodes). The extracted feature vectors for these tweets were used to train a C4.5 decision tree classifier (Hall et al., 2009). This classifier achieved an accuracy of 83.6% during training. We used this classifier and the hand-labeled set to seed the generic Yarowsky Algorithm (Abney, 2004), iteratively inducing a habit or episode label for all the tweets in the collection, using the WEKA output for confidence scoring and a confidence threshold of 0.96.

The extracted corpus was classified into 94,643 habitual tweets and 295,918 episodic tweets. To estimate the accuracy of the classifier, 400 randomly chosen tweets from the extracted corpus were hand-labeled, giving an estimated accuracy of 85% accuracy with 95% confidence, using the two-tailed t-test for sample size of proportions (p=0.05, n=400).

4.2 Results

Clearly the data in Figure 1 represents two combined distributions: one for episodes and one for habits, as we illustrate in Figure 2. We see that the verb *search* describes episodes that most often last minutes or hours, while it describes habits that go on for years.

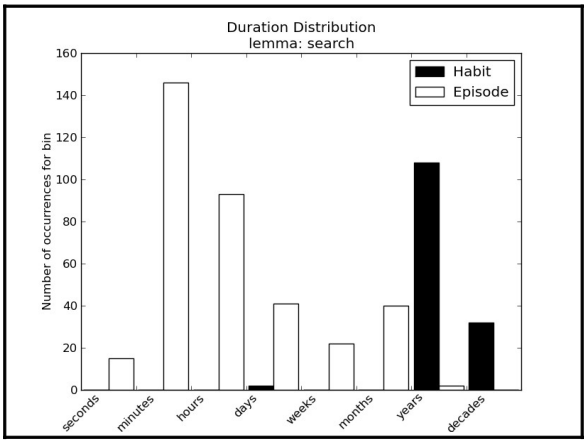


Figure 2: Duration distribution for *search*

These two different uses are illustrated in (4) and (5).

(4) *Obviously I'm the one who found the tiny lost black Lego in 30 seconds after the 3 of them searched for 5 minutes.*

(5) *@jaynecheeseman they've been searching for you for 11 years now. I'd look out if I were you.*

In Table 1 we provide summary information for several verb lemmas, indicating the average duration for each verb and the temporal unit corresponding to the largest bin for each verb.

Verb	Episodic Use		Habitual Use	
	Modal bin	**Mean**	**Modal bin**	**Mean**
snooze	minutes	1.6 hrs	decades	7.5 yrs
coach	hours	10 days	years	8.5 yrs
approve	minutes	1.7 mon.	years	1.4 yrs
eat	minutes	5.3 wks	days	5.7 yrs
kiss	seconds	4.5 days	weeks	1.8 yrs
visit	weeks	7.2 wks.	years	4.9 yrs

Table 1. Mean duration and mode for 6 of the verbs

It is clear that the methodology overestimates the duration of episodes somewhat – our estimates of typical durations are 2-3 times as long as those that come from the annotation in Pan, et. al. (2009). Nevertheless, the modal bin corresponds approximately to that the hand annotation in Pan, et. al., (2011) for nearly half (45%) of the verbs lemmas.

5 Conclusion

We have presented a hybrid approach for extracting typical durations of habits and episodes. We are able to extract high-quality information about temporal durations and to effectively classify tweets as to their habituality. It is clear that Twitter tweets contain a lot of unique data about different kinds of events and habits, and mining this data for temporal duration information has turned out to be a fruitful avenue for collecting the kind of world-knowledge that we need for robust temporal language processing. Our verb lexicon is available at: https://sites.google.com/site/relinguistics/.

References

Steven Abney. 2004. “Understanding the Yarowsky Algorithm”. *Computational Linguistics* 30(3): 365-395.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*.

Andrey Gusev, Nathaniel Chambers, Pranav Khaitan, Divye Khilnani, Steven Bethard, and Dan Jurafsky. 2011. “Using query patterns to learn the durations of events”. *IEEE IWCS-2011, 9th International Conference on Web Services.* Oxford, UK 2011.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

Rune Halvorsen, and Christopher Schierkolk. 2010. Tweetstream: Simple Twitter Streaming API (Version 0.3.5) [Software]. Available from https://bitbucket.org/runeh/tweetstream/src/.

Jerry Hobbs and James Pustejovsky. 2003. “Annotating and reasoning about time and events”. In *Proceedings of the AAAI Spring Symposium on Logical Formulation of Commonsense Reasoning*. Stanford University, CA 2003.

Zornitsa Kozareva and Eduard Hovy. 2011. “Learning Temporal Information for States and Events”. In *Proceedings of the Workshop on Semantic Annotation for Computational Linguistic Resources (ICSC 2011)*, Stanford.

Thomas Mathew and Graham Katz. 2009. “Supervised Categorization of Habitual and Episodic Sentences”. *Sixth Midwest Computational Linguistics Colloquium.* Bloomington, Indiana: Indiana University.

Marc Moens and Mark Steedman. 1988. “Temporal Ontology and Temporal Reference”. *Computational Linguistics* 14(2):15-28.

Feng Pan, Rutu Mulkar-Mehta, and Jerry R. Hobbs. 2006. “An Annotated Corpus of Typical Durations of Events”. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC)*, 77-82. Genoa, Italy.

Feng Pan, Rutu Mulkar-Mehta, and Jerry R. Hobbs. 2011. "Annotating and Learning Event Durations in Text." *Computational Linguistics* 37(4):727-752.

Event Linking: Grounding Event Reference in a News Archive

Joel Nothman[⊕] and **Matthew Honnibal**[+] and **Ben Hachey**[#] and **James R. Curran**[⊕]

[⊕]ə-lab, School of IT
University of Sydney
NSW, Australia

[⊕]Capital Markets CRC
55 Harrington St
Sydney
NSW, Australia

[+]Department of Computing
Macquarie University
NSW, Australia

[#]R&D, Thomson Reuters Corporation
St. Paul
MN, USA

{joel,james}@it.usyd.edu.au

{honnibal,ben.hachey}@gmail.com

Abstract

Interpreting news requires identifying its constituent events. Events are complex linguistically and ontologically, so disambiguating their reference is challenging. We introduce *event linking*, which canonically labels an event reference with the article where it was first reported. This implicitly relaxes coreference to *co-reporting*, and will practically enable augmenting news archives with semantic hyperlinks. We annotate and analyse a corpus of 150 documents, extracting 501 links to a news archive with reasonable inter-annotator agreement.

1 Introduction

Interpreting news requires identifying its constituent events. Information extraction (IE) makes this feasible by considering only events of a specified type, such as *personnel succession* or *arrest* (Grishman and Sundheim, 1996; LDC, 2005), an approach not extensible to novel events, or the same event types in sub-domains, e.g. sport. On the other hand, topic detection and tracking (TDT; Allan, 2002) disregards individual event mentions, clustering together articles that share a topic.

Between these fine and coarse-grained approaches, event identification requires grouping references to the same event. However, strict coreference is hampered by the complexity of event semantics: poison, murder and die may indicate the same effective event. The solution is to tag mentions with a canonical identifier for each news-triggering event.

This paper introduces *event linking*: given a past event reference in context, find the article in a news archive that first reports that the event happened. The task has an immediate practical application: some online newspapers link past event mentions to relevant news stories, but currently do so with low coverage and consistency; an event linker can add referentially-precise hyperlinks to news.

The event linking task parallels entity linking (NEL; Ji and Grishman, 2011), considering a news archive as a knowledge base (KB) of events, where each article exclusively represents the zero or more events that it first reports. Coupled with an appropriate event extractor, event linking may be performed for all events mentioned in a document, like the named entity disambiguation task (Bunescu and Paşca, 2006; Cucerzan, 2007).

We have annotated and analysed 150 news and opinion articles, marking references to past, newsworthy events, and linking where possible to canonical articles in a 13-year news archive.

2 The events in a news story

Approaches to news event processing are subsumed within broader notions of topics, scenario templates, or temporal entities, among others. We illustrate key challenges in processing news events and motivate event linking through the example story in Figure 1.

Salience Our story highlights carjackings and a police warning as newsworthy, alongside events like feeding, drove and told which carry less individual weight. Orthogonally, parts of the story are new events, while others are previously reported events that the reader may be aware of (illustrated in Figure 1). Online, the two background carjackings and the police warning are hyperlinked to other SMH articles where they were reported. Event schemas tend not to directly address salience: MUC-style IE (Gr-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 228–232,
Jeju, Republic of Korea, 8-14 July 2012.

N	**Sydney man carjacked at knifepoint**
B	There has been another carjacking in Sydney, two weeks after two people were stabbed in their cars in separate incidents.
N	A 32-year-old driver was walking to his station wagon on Hickson Road, Millers Point, after feeding his parking meter about 4.30pm yesterday when a man armed with a knife grabbed him and told him to hand over his car keys and mobile phone, police said. The carjacker then drove the black 2008 Holden Commodore... He was described as a 175-centimetre-tall Caucasian...
B	Police warned Sydney drivers to keep their car doors locked after two stabbings this month. On September 4, a 40-year-old man was stabbed when three men tried to steal his car on Rawson Street, Auburn, about 1.20am. The next day, a 25-year-old woman was stabbed in her lower back as she got into her car on Liverpool Road...

Figure 1: Possible event mentions marked in an article from SMH, segmented into news (N) and background (B) event portions.

ishman and Sundheim, 1996) selects an event type of which all instances are salient; TDT (Allan, 2002) operates at the document level, which avoids differentiating event mentions; and TimeML (Pustejovsky et al., 2003) marks the main event in each sentence. Critiquing ACE05 event detection for not addressing salience, Ji et al. (2009) harness cross-document frequencies for event ranking. Similarly, reference to a previously-reported event implies it is newsworthy.

Diversity IE traditionally targets a selected event type (Grishman and Sundheim, 1996). ACE05 considers a broader event typology, dividing eight thematic event types (*business*, *justice*, etc.) into 33 subtypes such as *attack*, *die* and *declare bankruptcy* (LDC, 2005). Most subtypes suffer from few annotated instances, while others are impractically broad: sexual abuse, gunfire and the Holocaust each constitute *attack* instances (is told considered an *attack* in Figure 1?). Inter-annotator agreement is low for most types.[1] While ACE05 would mark the various *attack* events in our story, police warned would be unrecognised. Despite template adaptation (Yangarber et al., 2000; Filatova et al., 2006; Li et al., 2010; Chambers and Jurafsky, 2011), event types are brittle to particular tasks and domains, such as bio-text mining (e.g. Kim et al., 2009); they cannot reasonably handle novel events.

[1]For binary sentence classification, we calculate an interquartile range of $\kappa \in [0.46, 0.64]$ over the 33 sub-types. Coarse event type classification ranges from $\kappa = 0.47$ for *business* to $\kappa = 0.69$ for *conflict*.

Identity Event coreference is complicated by partitive (sub-event) and logical (e.g. causation) relationships between events, in addition to lexical-semantic and syntactic issues. When considering the relationship between another carjacking and grabbed, drove or stabbed, ACE05 would apply the policy: "When in doubt, do not mark any coreference" (LDC, 2005). Bejan and Harabagiu (2008) consider event coreference across documents, marking the "most important events" (Bejan, 2010), albeit within Google News clusters, where multiple articles reporting the same event are likely to use similar language. Similar challenges apply to identifying event causality and other relations: Bejan and Harabagiu (2008) suggest arcs such as feeding $\xrightarrow{\text{precedes}}$ walking $\xrightarrow{\text{enables}}$ grabbed – akin to instantiations of FrameNet's frame relations (Fillmore et al., 2003). However, these too are semantically subtle.

Explicit reference By considering events through topical document clusters, TDT avoids some challenges of precise identity. It prescribes *rules of interpretation* for which stories pertain to a seminal event. However, the carjackings in our story are neither preconditions nor consequences of a seminal event and so would not constitute a TDT cluster. TDT fails to account for these explicit event references. Though Feng and Allan (2009) and Yang et al. (2009) consider event dependency as directed arcs between documents or paragraphs, they generally retain a broad sense of topic with little attention to explicit reference.

3 The event linking task

Given an explicit reference to a past event, event linking grounds it in a given news archive. This applies to all events worthy of having been reported, and harnesses explicit reference rather than more general notions of relevance. Though analogous to NEL, our task differs in the types of expressions that may be linked, and the manner of determining the correct KB node to link to, if any.

3.1 Event-referring expressions

We consider a subset of newsworthy events – *things that happen and directly trigger news* – as candidate referents. In TimeML's event classification (Pustejovsky et al., 2003), newsworthy events would gen-

erally be *occurrence* (e.g. die, build, sell) or *aspectual* (e.g. begin, discontinue), as opposed to *perception* (e.g. hear), *intentional state* (e.g. believe), etc. Still, we are not confined to these types when other classes of event are newsworthy. All references must be explicit, reporting the event as factual and completed or ongoing.

Not all event references meeting these criteria are reasonably LINKABLE to a single article:

MULTIPLE many distinct events, or an event type, e.g. world wars, demand;

AGGREGATE emerges from other events over time, e.g. grew 15%, scored 100 goals;

COMPLEX an event reported over multiple articles in terms of its sub-events, e.g. 2012 election, World Cup, scandal.

3.2 A news archive as a KB

We define a canonical link target for each event: *the earliest article in the archive that reports the given event happened or is happening*. Each archival article implicitly represents zero or more related events, just as Wikipedia entries represent zero or one entity in NEL. Links target the story as a whole: closely related, *co-reported* events link to the same article, avoiding a problematically strict approach to event identity. An archive reports only selected events, so a valid target may not exist (NEL's NIL).

4 An annotated corpus

We link to a digital archive of the Sydney Morning Herald: Australian and international news from 1986 to 2009, published daily, Monday to Saturday.[2] We annotate a randomly sampled corpus of 150 articles from its 2009 *News and Features* and *Business* sections including news reports, op-eds and letters.

For this whole-document annotation, a single word of each past/ongoing, newsworthy event mention is marked.[3] If LINKABLE, the annotator searches the archive by keyword and date, selecting a target, *reported here* (a self-referential link) or NIL. An annotation of our example story (Figure 1) would produce five groups of event references (Table 1).

[2]The archive may be searched at `http://newsstore.smh.com.au/apps/newsSearch.ac`

[3]We couple marking and linking since annotators must learn to judge newsworthiness relative to the target archive.

Mentions	Annotation category / link
carjacking; grabbed [him]	LINKABLE, reported here
[were] stabbed; incidents; stabbings	MULTIPLE
[Police] warned	LINKABLE, linked: *Sydney drivers told: lock your doors*
[man] stabbed	LINKABLE, linked: *Driver stabbed after Sydney carjacking*
[woman] stabbed	LINKABLE, linked: *Car attack: Driver stabbed in the back*

Table 1: Event linking annotations for Figure 1

Agreement unit	AB	AC	JA	JB	JC
Token has a link	27	21	61	42	34
Link target on agreed token	48	73	84	83	74
Set of link targets per document	31	40	69	51	45
Link date on agreed token	61	80	87	93	89
Set of link dates per document	36	44	71	54	56

Table 2: Inter-annotator and adjudicator F_1 scores

All documents were annotated by external annotator A; external annotators B and C annotated 72 and 24 respectively; and all were adjudicated by the first author (J). Pairwise inter-annotator agreement in Table 2 shows that annotators infrequently select the same words to link, but that reasonable agreement on the link target can be achieved for agreed tokens.[4] Adjudicator-annotator agreements are generally much higher than inter-annotator agreements: in many cases, an annotator fails to find a target or selects one that does not first report the event; J accepts most annotations as valid. In other cases, there may be multiple articles published on the same day that describe the event in question from different angles; agreement increases substantially when relaxed to accept date agreement. Our adjudicated corpus of 150 documents is summarised in Table 3.

Where a definitive link target is not available, an annotator may erroneously select another candidate: an opinion article describing the event, an article where the event is mentioned as background, or an article anticipating the event.

The task is complicated by changed perspective between an event's first report and its later reference.

[4]$\kappa \approx F_1$ for the binary token task (F_1 accounts for the majority class) and for the sparse link targets/date selection.

Category	Mentions	Types	Docs
Any markable	2136	655	149
LINKABLE	1399	417	144
linked	501	229	99
reported here	667	111	111
nil	231	77	77
COMPLEX	220	79	79
MULTIPLE	328	102	102
AGGREGATE	189	57	57

Table 3: Annotation frequencies: no. of mentions, distinct per document, and document frequency

Can overpayed link to what had been acquired? Can 10 died be linked to an article where only nine are confirmed dead? For the application of adding hyperlinks to news, such a link might be beneficial, but it may be better considered an AGGREGATE.

The schema underspecifies definitions of 'event' and 'newsworthiness', accounting for much of the token-level disagreement, but not directly affecting the task of linking a specified mention to the archive. Adjectival mentions such as Apple's *new* CEO are easy to miss and questionably explicit. Events are also confused with facts and abstract entities, such as bans, plans, reports and laws. Unlike many other facts, events can be grounded to a particular time of occurrence, often stated in text.

5 Analysis and discussion

To assess task feasibility, we present bag-of-words (BoW) and oracle results (Figure 2). Using the whole document as a query[5] retrieves 30% of gold targets at rank 10, but only 60% by rank 150. Term windows around each event mention perform close to our oracle consisting of successful search keywords collected during annotation, with over 80% recall at 150. No system recalls over 30% of targets at 1-best, suggesting a reranking approach may be required.

Constraining search result dates is essential; annotators' constraints improve recall by 20% at rank 50. These constraints may draw on temporal expressions in the source article or external knowledge. Successful automated linking will therefore require extensive use of semantic and temporal information.

Our corpus also highlights distinctions between

[5]Using Apache Solr defaults: TFIDF-weighted cosine similarity over stemmed and stopped tokens.

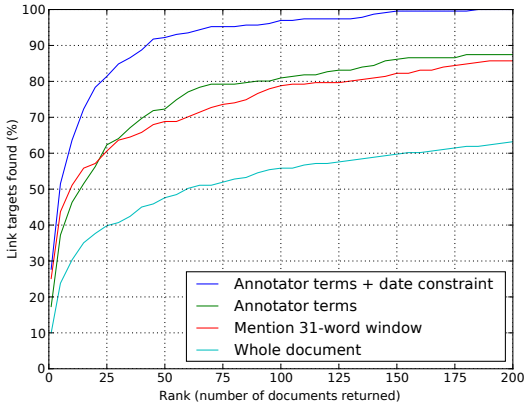


Figure 2: Recall for BoW and oracle systems

explicit event reference and broader relationships. Yang et al. (2009) makes the reasonable assumption that news events generally build on others that recently precede them. We find that the likelihood a linked article occurred fewer than d days ago reduces exponentially with respect to d, yet the rate of decay is surprisingly slow: half of all link targets precede their source by over 3 months.

The effect of coreporting rather than coreference is also clear: like {carjacking, grabbed} in our example, mention chains include {return, decide, recontest}, {winner, Cup} as well as more familiar instances like {acquired, acquisition}.

6 Conclusion

We have introduced *event linking*, which takes a novel approach to news event reference, associating each newsworthy past event with a canonical article in a news archive. We demonstrate task's feasibility, with reasonable inter-annotator agreement over a 150 document corpus. The corpus highlights features of the retrieval task and its dependence on temporal knowledge. As well as using event linking to add referentially precise hyperlinks to a news archive, further characteristics of news will emerge by analysing the graph of event references.

7 Acknowledgements

We are grateful to the reviewers for their comments. The work was supported by Capital Markets CRC post-doctoral fellowships (BH; MH) and PhD Scholarship (JN); a University of Sydney VCRS (JN); and ARC Discovery Grant DP1097291 (JRC).

References

James Allan, editor. 2002. *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers, Boston, MA.

Cosmin Adrian Bejan and Sanda Harabagiu. 2008. A linguistic resource for discovering event structures and resolving event coreference. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco.

Cosmin Adrian Bejan. 2010. Private correspondence, November.

Razvan Bunescu and Marius Paşca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 9–16.

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*, pages 976–986, Portland, Oregon, USA, June.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 708–716.

Ao Feng and James Allan. 2009. Incident threading for news passages. In *CIKM '09: Proceedings of the 18th ACM international conference on Information and knowledge management*, pages 1307–1316, Hong Kong, November.

Elena Filatova, Vasileios Hatzivassiloglou, and Kathleen McKeown. 2006. Automatic creation of domain templates. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 207–214, Sydney, Australia, July.

Charles J. Fillmore, Christopher R. Johnson, and Miriam R. L. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16(3):235–250.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference – 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, June.

Heng Ji, Ralph Grishman, Zheng Chen, and Prashant Gupta. 2009. Cross-document event extraction and tracking: Task, evaluation, techniques and challenges. In *Proceedings of Recent Advances in Natural Language Processing*, September.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado, June.

LDC. 2005. ACE (Automatic Content Extraction) English annotation guidelines for events. Linguistic Data Consortium, July. Version 5.4.3.

Hao Li, Xiang Li, Heng Ji, and Yuval Marton. 2010. Domain-independent novel event discovery and semi-automatic event annotation. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, Sendai, Japan, November.

James Pustejovsky, José Casta no, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003. TimeML: Robust specification of event and temporal expressions in text. In *Proceedings of the Fifth International Workshop on Computational Semantics*.

Christopher C. Yang, Xiaodong Shi, and Chih-Ping Wei. 2009. Discovering event evolution graphs from news corpora. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 34(4):850–863, July.

Roman Yangarber, Ralph Grishman, and Pasi Tapanainen. 2000. Automatic acquisition of domain knowledge for information extraction. In *In Proceedings of the 18th International Conference on Computational Linguistics*, pages 940–946.

Coupling Label Propagation and Constraints for Temporal Fact Extraction

Yafang Wang, Maximilian Dylla, Marc Spaniol and Gerhard Weikum
Max Planck Institute for Informatics, Saarbrücken, Germany
{ywang|mdylla|mspaniol|weikum}@mpi-inf.mpg.de

Abstract

The Web and digitized text sources contain a wealth of information about named entities such as politicians, actors, companies, or cultural landmarks. Extracting this information has enabled the automated construction of large knowledge bases, containing hundred millions of binary relationships or attribute values about these named entities. However, in reality most knowledge is transient, i.e. changes over time, requiring a temporal dimension in fact extraction. In this paper we develop a methodology that combines label propagation with constraint reasoning for temporal fact extraction. Label propagation aggressively gathers fact candidates, and an Integer Linear Program is used to clean out false hypotheses that violate temporal constraints. Our method is able to improve on recall while keeping up with precision, which we demonstrate by experiments with biography-style Wikipedia pages and a large corpus of news articles.

1 Introduction

In recent years, automated fact extraction from Web contents has seen significant progress with the emergence of freely available knowledge bases, such as DBpedia (Auer et al., 2007), YAGO (Suchanek et al., 2007), TextRunner (Etzioni et al., 2008), or ReadTheWeb (Carlson et al., 2010a). These knowledge bases are constantly growing and contain currently (by example of DBpedia) several million entities and half a billion facts about them. This wealth of data allows to satisfy the information needs of advanced Internet users by raising queries from keywords to entities. This enables queries like "Who is married to Prince Charles?" or "Who are the teammates of Lionel Messi at FC Barcelona?".

However, factual knowledge is highly ephemeral: Royals get married and divorced, politicians hold positions only for a limited time and soccer players transfer from one club to another. Consequently, knowledge bases should be able to support more sophisticated *temporal* queries at *entity-level*, such as "Who have been the spouses of Prince Charles before 2000?" or "Who are the teammates of Lionel Messi at FC Barcelona in the season 2011/2012?". In order to achieve this goal, the next big step is to distill *temporal knowledge* from the Web.

Extracting temporal facts is a complex and time-consuming endeavor. There are "conservative" strategies that aim at high precision, but they tend to suffer from low recall. On the contrary, there are "aggressive" approaches that target at high recall, but frequently suffer from low precision. To this end, we introduce a method that allows us to gain maximum benefit from both "worlds" by "aggressively" gathering fact candidates and subsequently "cleaning-up" the incorrect ones. The salient properties of our approach and the novel contributions of this paper are the following:

- A temporal fact extraction strategy that is able to efficiently gather thousands of fact candidates based on a handful of seed facts.
- An ILP solver incorporating constraints on temporal relations among events (e.g., marriage of a person must be non-overlapping in time).
- Experiments on real world news and Wikipedia articles showing that we gain recall while keeping up with precision.

2 Related Work

Recently, there have been several approaches that aim at the extraction of temporal facts for the automated construction of large knowledge bases, but

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 233–237,
Jeju, Republic of Korea, 8-14 July 2012.

time-aware fact extraction is still in its infancy. An approach toward fact extraction based on coupled semi-supervised learning for information extraction (IE) is NELL (Carlson et al., 2010b). However, it does neither incorporate constraints nor temporality. TIE (Ling and Weld, 2010) binds time-points of events described in sentences, but does not disambiguate entities or combine observations to facts. A pattern-based approach for temporal fact extraction is PRAVDA (Wang et al., 2011), which utilizes label propagation as a semi-supervised learning strategy, but does not incorporate constraints. Similarly, TOB is an approach of extracting temporal business-related facts from free text, which requires deep parsing and does not apply constraints as well (Zhang et al., 2008). In contrast, CoTS (Talukdar et al., 2012) introduces a constraint-based approach of coupled semi-supervised learning for IE, however not focusing on the extraction part. Building on TimeML (Pustejovsky et al., 2003) several works (Verhagen et al., 2005; Mani et al., 2006; Chambers and Jurafsky, 2008; Verhagen et al., 2009; Yoshikawa et al., 2009) identify temporal relationships in free text, but don't focus on fact extraction.

3 Framework

Facts and Observations. We aim to extract factual knowledge transient over time from free text. More specifically, we assume *time* $\mathcal{T} = [0, T_{max}]$ to be a finite sequence of time-points with yearly granularity. Furthermore, a *fact* consists of a relation with two typed arguments and a time-interval defining its validity. For instance, we write $worksForClub(Beckham, RMadrid)@[2003, 2008)$ to express that Beckham played for Real Madrid from 2003 to 2007. Since sentences containing a fact and its full time-interval are sparse, we consider three kinds of textual observations for each relation, namely *begin*, *during*, and *end*. "Beckham signed for Real Madrid from Manchester United in 2003." includes both the *begin* observation of Beckham being with Real Madrid as well as the *end* observation of working for Manchester. A *positive seed fact* is a valid fact of a relation, while a *negative seed fact* is incorrect (e.g., for relation $worksForClub$, a *positive seed fact* is $worksForClub(Beckham, RMadrid)$, while $worksForClub(Beckham, BMunich)$ is a *negative seed fact*).

Framework. As depicted in Figure 1, our framework is composed of four stages, where the first collects candidate sentences, the second mines patterns from the candidates sentences, the third extracts temporal facts from the sentences utilizing the patterns and the last removes noisy facts by enforcing constraints.

Preprocessing. We retrieve all sentences from the corpus comprising at least two entities and a temporal expression, where we use YAGO for entity recognition and disambiguation (cf. (Hoffart et al., 2011)).

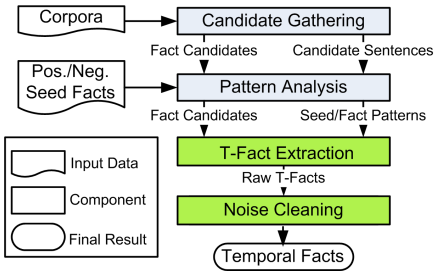


Figure 1: System Overview

Pattern Analysis. A *pattern* is a n-gram based feature vector. It is generated by replacing entities by their types, keeping only stemmed nouns, verbs converted to present tense and the last preposition. For example, considering "Beckham signed for Real Madrid from Manchester United in 2003." the corresponding pattern for the *end* occurrence is "sign for CLUB from". We quantify the *strength* of each pattern by investigating how frequent the pattern occurs with seed facts of a particular relation and how infrequent it appears with negative seed facts.

Fact Candidate Gathering. Entity pairs that co-occur with patterns whose strength is above a minimum threshold become fact candidates and are fed into the next stage of label propagation.

4 T-Fact Extraction

Building on (Wang et al., 2011) we utilize Label Propagation (Talukdar and Crammer, 2009) to determine the relation and observation type expressed by each pattern.

Graph. We create a graph $G = (\mathcal{V}_F \dot{\cup} \mathcal{V}_P, \mathcal{E})$ having one vertex $v \in \mathcal{V}_F$ for each fact candidate observed in the text and one vertex $v \in \mathcal{V}_P$ for each pattern. Edges between $\mathcal{V}_F$ and $\mathcal{V}_P$ are introduced whenever a fact candidate appeared with a pattern. Their weight is derived from the co-occurrence frequency. Edges

among $\mathcal{V}_P$ nodes have weights derived from the n-gram overlap of the patterns.

Labels. Moreover, we use one label for each observation type (*begin*, *during*, and *end*) of each relation and a dummy label representing the unknown relation.

Objective Function. Let $\mathbf{Y} \in \mathbb{R}_+^{|\mathcal{V}|\times|Labels|}$ denote the graph's initial label assignment, and $\widehat{\mathbf{Y}} \in \mathbb{R}_+^{|\mathcal{V}|\times|Labels|}$ stand for the estimated labels of all vertices, $\mathbf{S}_l$ encode the seed's weights on its diagonal, and $\mathbf{R}_{*l}$ contain zeroes except for the dummy label's column. Then, the objective function is:

$$\mathcal{L}(\widehat{\mathbf{Y}}) = \sum_{\ell} \begin{bmatrix} (\mathbf{Y}_{*\ell} - \widehat{\mathbf{Y}}_{*\ell})^T \mathbf{S}_\ell (\mathbf{Y}_{*\ell} - \widehat{\mathbf{Y}}_{*\ell}) \\ +\mu_1 \widehat{\mathbf{Y}}_{*\ell}^T \mathbf{L} \widehat{\mathbf{Y}}_{*\ell} + \mu_2 \|\widehat{\mathbf{Y}}_{*\ell} - \mathbf{R}_{*\ell}\|^2 \end{bmatrix} \quad (1)$$

Here, the first term $(\mathbf{Y}_{*\ell} - \widehat{\mathbf{Y}}_{*\ell})^T \mathbf{S}_\ell (\mathbf{Y}_{*\ell} - \widehat{\mathbf{Y}}_{*\ell})$ ensures that the estimated labels approximate the initial labels. The labeling of neighboring vertices is smoothed by $\mu_1 \widehat{\mathbf{Y}}_{*\ell}^T \mathbf{L} \widehat{\mathbf{Y}}_{*\ell}$, where $\mathbf{L}$ refers to the Laplacian matrix. The last term is a L2 regularizer.

5 Cleaning of Fact Candidates

To prune noisy t-facts, we compute a consistent subset of t-facts with respect to temporal constraints (e.g. joining a sports club takes place before leaving a sports club) by an Integer Linear Program (ILP).

Variables. We introduce a variable $x_r \in \{0, 1\}$ for each t-fact candidate $r \in \mathcal{R}$, where 1 means the candidate is valid. Two variables $x_{f,b}, x_{f,e} \in [0, T_{max}]$ denote begin (b) and end (e) of time-interval of a fact $f \in \mathcal{F}$. Note, that many t-fact candidates refer to the same fact f, since they share their entity pairs.

Objective Function. The objective function intends to maximize the number of valid raw t-facts, where w_r is a weight obtained from the previous stage:

$$\max \sum_{r \in \mathcal{R}} w_r \cdot x_r$$

Intra-Fact Constraints. $x_{f,b}$ and $x_{f,e}$ encode a proper time-interval by adding the constraint:

$$\forall f \in \mathcal{F} \quad x_{f,b} < x_{f,e}$$

Considering only a single relation, we assume the sets $\mathcal{R}_b$, $\mathcal{R}_d$, and $\mathcal{R}_e$ to comprise its t-fact candidates with respect to the *begin*, *during*, and *end* observations. Then, we introduce the constraints

$$\forall l \in \{b, e\}, r \in \mathcal{R}_l \quad t_l \cdot x_r \leq x_{f,l} \quad (2)$$

$$\forall l \in \{b, e\}, r \in \mathcal{R}_l \quad x_{f,l} \leq t_l \cdot x_r + (1 - x_r) T_{max} \quad (3)$$

$$\forall r \in \mathcal{R}_d \quad x_{f,b} \leq t_b \cdot x_r + (1 - x_r) T_{max} \quad (4)$$

$$\forall r \in \mathcal{R}_d \quad t_e \cdot x_r \leq x_{f,e} \quad (5)$$

where f has the same entity pair as r and t_b, t_e are begin and end of r's time-interval. Whenever x_r is set to 1 for *begin* or *end* t-fact candidates, Eq. (2) and Eq. (3) set the value of $x_{f,b}$ or $x_{f,e}$ to t_b or t_e, respectively. For each *during* t-fact candidate with $x_r = 1$, Eq. (4) and Eq. (5) enforce $x_{f,b} \leq t_b$ and $t_e \leq x_{f,e}$.

Inter-Fact Constraints. Since we can refer to a fact f's time interval by $x_{f,b}$ and $x_{f,e}$ and the connectives of Boolean Logic can be encoded in ILPs (Karp, 1972), we can use all temporal constraints expressible by Allen's Interval Algebra (Allen, 1983) to specify inter-fact constraints. For example, we leverage this by prohibiting marriages of a single person from overlapping in time.

Previous Work. In comparison to (Talukdar et al., 2012), our ILP encoding is time-scale invariant. That is, for the same data, if the granularity of $\mathcal{T}$ is changed from months to seconds, for example, the size of the ILP is not affected. Furthermore, because we allow all relations of Allen's Interval Algebra, we support a richer class of temporal constraints.

6 Experiments

Corpus. Experiments are conducted in the soccer and the celebrity domain by considering the *worksForClub* and *isMarriedTo* relation, respectively. For each person in the "FIFA 100 list" and "Forbes 100 list" we retrieve their Wikipedia article. In addition, we obtained about 80,000 documents for the soccer domain and 370,000 documents for the celebrity domain from BBC, The Telegraph, Times Online and ESPN by querying Google's News Archive Search[1] in the time window from 1990-2011. All hyperparameters are tuned on a separate data-set.

Seeds. For each relation we manually select the 10 positive and negative fact candidates with highest occurrence frequencies in the corpus as seeds.

Evaluation. We evaluate *precision* by randomly sampling 50 (*isMarriedTo*) and 100 (*worksForClub*) facts for each observation type and manually evaluating them against the text documents. All experimental data is available for download from our website[2].

6.1 Pipeline vs. Joint Model

Setting. In this experiment we compare the performance of the pipeline being stages 3 and 4 in Figure

[1] news.google.com/archivesearch

[2] www.mpi-inf.mpg.de/yago-naga/pravda/

1 and a joint model in form of an ILP solving the t-fact extraction and noise cleaning at the same time. Hence, the joint model resembles (Roth and Yih, 2004) extended by Section 5's temporal constraints.

Relation	Observation	Label Propagation		ILP for T-Fact Extraction		
		Precision	# Obs.	Precision	# Obs.	
worksForClub	begin	80%	2537	81%	2426	Without Noise Cleaning
	during	78%	2826	86%	1153	
	end	65%	440	50%	550	
isMarriedTo	begin	52%	195	28%	232	
	during	76%	92	6%	466	
	end	62%	50	2%	551	
worksForClub	begin	85%	2469	87%	2076	With Noise Cleaning
	during	85%	2761	79%	1434	
	end	74%	403	72%	275	
isMarriedTo	begin	64%	177	74%	67	
	during	79%	89	88%	61	
	end	70%	47	71%	28	

Table 1: Pipeline vs. Joint Model

Results. Table 1 shows the results on the pipeline model (lower-left), joint model (lower-right), label-propagation w/o noise cleaning (upper-left), and ILP for t-fact extraction w/o noise cleaning (upper-right).
Analysis. Regarding the upper part of Table 1 the pattern-based extraction works very well for *worksForClub*, however it fails on *isMarriedTo*. The reason is, that the types of *worksForClub* distinguish the patterns well from other relations. In contrast, *isMarriedTo*'s patterns interfere with other person-person relations making constraints a decisive asset. When comparing the joint model and the pipeline model, the former sacrifices recall in order to keep up with the latter's precision level. That is because the joint model's ILP decides with binary variables on which patterns to accept. In contrast, label propagation addresses the inherent uncertainty by providing label assignments with confidence numbers.

6.2 Increasing Recall

Setting. In a second experiment, we move the t-fact extraction stage away from high precision towards higher recall, where the successive noise cleaning stage attempts to restore the precision level.
Results. The columns of Table 2 show results for different values of μ_1 of Eq. (1). From left to right, we used $\mu_1 = e^{-1}, 0.6, 0.8$ for *worksForClub* and $\mu_1 = e^{-2}, e^{-1}, 0.6$ for *isMarriedTo*. The table's upper part reports on the output of stage 3, whereas the lower part covers the facts returned by noise cleaning.
Analysis. For the conservative setting label propagation produces high precision facts with only few inconsistencies, so the noise cleaning stage has no effect, i.e. no pruning takes place. This is the setting usual pattern-based approaches without cleaning stage are working in. In contrast, for the standard setting (coinciding with Table 1's left column) stage 3 yields less precision, but higher recall. Since there are more inconsistencies in this setup, the noise cleaning stage accomplishes precision gains compensating for the losses in the previous stage. In the relaxed setting precision drops too low, so the noise cleaning stage is unable to figure out the truly correct facts. In general, the effects on *worksForClub* are weaker, since in this relation the constraints are less influential.

		Conservative		Standard		Relaxed		
		Prec.	# Obs.	Prec.	# Obs.	Prec.	# Obs.	
worksForClub	begin	83%	2443	80%	2537	80%	2608	Without Noise Cleaning
	during	81%	2523	78%	2826	76%	2928	
	end	77%	377	65%	440	62%	501	
isMarriedTo	begin	72%	112	52%	195	44%	269	
	during	90%	63	76%	92	52%	187	
	end	67%	37	62%	50	36%	116	
worksForClub	begin	83%	2389	85%	2469	84%	2536	With Noise Cleaning
	during	88%	2474	85%	2761	75%	2861	
	end	79%	349	72%	403	70%	463	
isMarriedTo	begin	72%	111	64%	177	46%	239	
	during	90%	62	79%	89	54%	177	
	end	69%	36	68%	47	38%	110	

Table 2: Increasing Recall.

7 Conclusion

In this paper we have developed a method that combines label propagation with constraint reasoning for temporal fact extraction. Our experiments have shown that best results can be achieved by applying "aggressive" label propagation with a subsequent ILP for "clean-up". By coupling both approaches we achieve both high(er) precision and high(er) recall. Thus, our method efficiently extracts high quality temporal facts at large scale.

Acknowledgements

This work is supported by the 7^{th} Framework IST programme of the European Union through the focused research project (STREP) on Longitudinal Analytics of Web Archive data (LAWA) under contract no. 258105.

References

James F. Allen. 1983. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, November.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *In 6th Intl Semantic Web Conference, Busan, Korea*, pages 11–15. Springer.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010a. Toward an architecture for never-ending language learning. In *AAAI*, pages 1306–1313.

Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010b. Coupled semi-supervised learning for information extraction. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM 2010)*.

Nathanael Chambers and Daniel Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *EMNLP*, pages 698–706.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Commun. ACM*, 51(12):68–74, December.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proc. of EMNLP 2011: Conference on Empirical Methods in Natural Language Processing, Edinburgh, Scotland, UK, July 2731*, pages 782–792.

Richard M. Karp. 1972. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103.

Xiao Ling and Daniel S. Weld. 2010. Temporal information extraction. In *Proceedings of the AAAI 2010 Conference*, pages 1385 – 1390, Atlanta, Georgia, USA, July 11-15. Association for the Advancement of Artificial Intelligence.

Inderjeet Mani, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. In *In ACL-06*, pages 17–18.

James Pustejovsky, José M. Castaño, Robert Ingria, Roser Sauri, Robert J. Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R. Radev. 2003. TimeML: Robust specification of event and temporal expressions in text. In *New Directions in Question Answering*, pages 28–34.

Dan Roth and Wen-Tau Yih, 2004. *A Linear Programming Formulation for Global Inference in Natural Language Tasks*, pages 1–8.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web*, pages 697–706, New York, NY, USA. ACM.

Partha Pratim Talukdar and Koby Crammer. 2009. New regularized algorithms for transductive learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, ECML PKDD '09, pages 442–457, Berlin, Heidelberg. Springer-Verlag.

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.

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 *ACL '05: Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 81–84, Morristown, NJ, USA. Association for Computational Linguistics.

Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Jessica Moszkowicz, and James Pustejovsky. 2009. The tempeval challenge: identifying temporal relations in text. *Language Resources and Evaluation*, 43:161–179.

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.

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 Suchanek, and Gerhard Weikum. 2008. TOB: Timely ontologies for business relations. In *11th International Workshop on Web and Databases 2008 (WebDB 2008)*, Vancouver, Canada. ACM.

Using Search-Logs to Improve Query Tagging

Kuzman Ganchev Keith Hall Ryan McDonald Slav Petrov
Google, Inc.
{kuzman|kbhall|ryanmcd|slav}@google.com

Abstract

Syntactic analysis of search queries is important for a variety of information-retrieval tasks; however, the lack of annotated data makes training query analysis models difficult. We propose a simple, efficient procedure in which part-of-speech tags are transferred from retrieval-result snippets to queries at training time. Unlike previous work, our final model does not require any additional resources at run-time. Compared to a state-of-the-art approach, we achieve more than 20% relative error reduction. Additionally, we annotate a corpus of search queries with part-of-speech tags, providing a resource for future work on syntactic query analysis.

1 Introduction

Syntactic analysis of search queries is important for a variety of tasks including better query refinement, improved matching and better ad targeting (Barr et al., 2008). However, search queries differ substantially from traditional forms of written language (e.g., no capitalization, few function words, fairly free word order, etc.), and are therefore difficult to process with natural language processing tools trained on standard corpora (Barr et al., 2008). In this paper we focus on part-of-speech (POS) tagging queries entered into commercial search engines and compare different strategies for learning from search logs. The search logs consist of user queries and relevant search results retrieved by a search engine. We use a supervised POS tagger to label the result snippets and then transfer the tags to the queries, producing a set of noisy labeled queries. These labeled queries are then added to the training data and the tagger is retrained. We evaluate different strategies for selecting which annotation to transfer and find that using the result that was clicked by the user gives comparable performance to using just the top result or to aggregating over the top-k results.

The most closely related previous work is that of Bendersky et al. (2010, 2011). In their work, unigram POS tag priors generated from a large corpus are blended with information from the top-50 results from a search engine at prediction time. Such an approach has the disadvantage that it necessitates access to a search engine at run-time and is computationally very expensive. We re-implement their method and show that our direct transfer approach is more effective, while being simpler to instrument: since we use information from the search engine only during training, we can train a stand-alone POS tagger that can be run without access to additional resources. We also perform an error analysis and find that most of the remaining errors are due to errors in POS tagging of the snippets.

2 Direct Transfer

The main intuition behind our work, Bendersky et al. (2010) and Rüd et al. (2011), is that standard NLP annotation tools work better on snippets returned by a search engine than on user supplied queries. This is because snippets are typically well-formed English sentences, while queries are not. Our goal is to leverage this observation and use a supervised POS tagger trained on regular English sentences to generate annotations for a large set of queries that can be used for training a query-specific model. Perhaps the simplest approach – but also a surprisingly powerful one – is to POS tag some relevant snippets for

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 238–242,
Jeju, Republic of Korea, 8-14 July 2012.

a given query, and then to transfer the tags from the snippet tokens to matching query tokens. This "direct" transfer idea is at the core of all our experiments. In this work, we provide a comparison of techniques for selecting snippets associated with the query, as well as an evaluation of methods for aligning the matching words in the query to those in the selected snippets.

Specifically, for each query[1] with a corresponding set of "relevant snippets," we first apply the baseline tagger to the query and all the snippets. We match any query terms in these snippets, and copy over the POS tag to the matching query term. Note that this can produce multiple labelings as the relevant snippet set can be very diverse and varies even for the same query. We choose the most frequent tagging as the canonical one and add it to our training set. We then train a query tagger on all our training data: the original human annotated English sentences and also the automatically generated query training set.

The simplest way to match query tokens to snippet tokens is to allow a query token to match any snippet token. This can be problematic when we have queries that have a token repeated with different parts-of-speech such as in "tie a tie." To make a more precise matching we try a sequence of matching rules: First, exact match of the query n-gram. Then matching the terms in order, so the query "tie$_a$ a tie$_b$" matched to the snippet "to tie$_1$ a neck tie$_2$" would match *tie$_a$:tie$_1$* and *tie$_b$:tie$_2$*. Finally, we match as many query terms as possible. An early observation showed that when a query term occurs in the result URL, e.g., searching for "irs mileage rate" results in the page `irs.gov`, the query term matching the URL domain name is usually a proper noun. Consequently we add this rule.

In the context of search logs, a relevant snippet set can refer to the top k snippets (including the case where $k = 1$) or the snippet(s) associated with results clicked by users that issued the query. In our experiments we found that different strategies for selecting relevant snippets, such as selecting the snippets of the clicked results, using the top-10 results or using only the top result, perform similarly (see Table 1).

[1]We skip navigational queries, e.g, *amazon* or *amazon.com*, since syntactic analysis of such queries is not useful.

Query	budget/NN rent/VB a/DET car/NN	Clicks
Snip 1	... Budget/NNP Rent/NNP A/NNP Car/NNP ...	2
Snip 2	... Go/VB to/TO Budget/NNP to/TO rent/VB a/DET car/NN ...	1
Snip 3	... Rent/VB a/DET car/NN from/IN Budget/NNP ...	1

Figure 1: Example query and snippets as tagged by a baseline tagger as well as associated clicks.

By contrast Bendersky et al. (2010) use a linear interpolation between a prior probability and the snippet tagging. They define $\pi(t|w)$ as the relative frequency of tag t given by the baseline tagger to word w in some corpus and $\psi(t|w, s)$ as the indicator function for word w in the context of snippet s has tag t. They define the tagging of a word as

$$\arg\max_t \; 0.2\pi(t|w) + 0.8 \operatorname*{mean}_{s:w\in s} \psi(t|w, s) \quad (1)$$

We illustrate the difference between the two approaches in Figure 1. The numbered rows of the table correspond to three snippets (with non-query terms elided). The strategy that uses the clicks to select the tagging would count two examples of "Budget/NNP Rent/NNP A/NNP Car/NNP" and one for each of two other taggings. Note that snippet 1 and the query get different taggings primarily due to orthographic variations. It would then add "budget/NNP rent/NNP a/NNP car/NNP" to its training set. The interpolation approach of Bendersky et al. (2010) would tag the query as "budget/NNP rent/VB a/DET car/NN". To see why this is the case, consider the probability for rent/VB vs rent/NNP. For rent/VB we have $0.2 + 0.8 \times \frac{2}{3}$, while for rent/NNP we have $0 + 0.8 \times \frac{1}{3}$ assuming that $\pi(\text{VB}|\text{rent}) = 1$.

3 Experimental Setup

We assume that we have access to labeled English sentences from the PennTreebank (Marcus et al., 1993) and the QuestionBank (Judge et al., 2006), as well as large amounts of unlabeled search queries. Each query is paired with a set of relevant results represented by snippets (sentence fragments containing the search terms), as well as information about the order in which the results were shown to the user and possibly the result the user clicked on. Note that different sets of results are possible for the

same query, because of personalization and ranking changes over time.

3.1 Evaluation Data

We use two data sets for evaluation. The first is the set of 251 queries from Microsoft search logs (MS-251) used in Bendersky et al. (2010, 2011). The queries are annotated with three POS tags representing nouns, verbs and "other" tags (MS-251 NVX). We additionally refine the annotation to cover 14 POS tags comprising the 12 universal tags of Petrov et al. (2012), as well as proper nouns and a special tag for search operator symbols such as "-" (for excluding the subsequent word). We refer to this evaluation set as MS-251 in our experiments. We had two annotators annotate the whole of the MS-251 data set. Before arbitration, the inter-annotator agreement was 90.2%. As a reference, Barr et al. (2008) report 79.3% when annotating queries with 19 POS tags. We then examined all the instances where the annotators disagreed, and corrected the discrepancy. Our annotations are available at `http://code.google.com/p/query-syntax/`.

The second evaluation set consists of 500 so called "long-tail" queries. These are queries that occurred rarely in the search logs, and are typically difficult to tag because they are searching for less-frequent information. They do not contain navigational queries.

3.2 Baseline Model

We use a linear chain tagger trained with the averaged perceptron (Collins, 2002). We use the following features for our tagger: current word, suffixes and prefixes of length 1 to 3; additionally we use word cluster features (Uszkoreit and Brants, 2008) for the current word, and transition features of the cluster of the current and previous word. When training on Sections 1-18 of the Penn Treebank and testing on sections 22-24, our tagger achieves 97.22% accuracy with the Penn Treebank tag set, which is state-of-the-art for this data set. When we evaluate only on the 14 tags used in our experiments, the accuracy increases to 97.88%.

We experimented with 4 baseline taggers (see Table 2). WSJ corresponds to training on only the standard training sections of Wall Street Journal portion of the Penn Treebank. WSJ+QTB adds the QuestionBank as training data. WSJ NOCASE and WSJ+QTB NOCASE use case-insensitive version of the tagger (conceptually lowercasing the text before training and before applying the tagger). As we will see, all our baseline models are better than the baseline reported in Bendersky et al. (2010); our lowercased baseline model significantly outperforms even their best model.

Method	MS-251 NVX	MS-251	long-tail
DIRECT-CLICK	93.43	84.11	78.15
DIRECT-ALL	93.93	84.39	77.73
DIRECT-TOP-1	93.93	84.60	77.60

Table 1: Evaluation of snippet selection strategies.

4 Experiments

First, we compared different strategies for selecting relevant snippets from which to transfer the tags. These systems are: DIRECT-CLICK, which uses snippets clicked on by users; DIRECT-ALL, which uses all the returned snippets seen by the user;[2] and DIRECT-TOP-1, which uses just the snippet in the top result. Table 1 compares these systems on our three evaluation sets. While DIRECT-ALL and DIRECT-TOP-1 perform best on the MS-251 data sets, DIRECT-CLICK has an advantage on the long tail queries. However, these differences are small ($<$0.6%) suggesting that any strategy for selecting relevant snippet sets will return comparable results when aggregated over large amounts of data.

We then compared our method to the baseline models and a re-implementation of Bendersky et al. (2010), which we denote BSC. We use the same matching scheme for both BSC and our system, including the URL matching described in Section 2. The URL matching improves performance by 0.4-3.0% across all models and evaluation settings.

Table 2 summarizes our final results. For comparison, Bendersky et al. (2010) report 91.6% for their final system, which is comparable to our implementation of their system when the baseline tagger is trained on just the WSJ corpus. Our best system achieves a 21.2% relative reduction in error on their annotations. Some other trends become appar-

[2]Usually 10 results, but more if the user viewed the second page of results.

Method	MS-251 NVX	MS-251	long-tail
WSJ	90.54	75.07	53.06
BSC	91.74	77.82	57.65
DIRECT-CLICK	93.36	**85.81**	76.13
WSJ + QTB	90.18	74.86	53.48
BSC	91.74	77.54	57.65
DIRECT-CLICK	93.01	85.03	76.97
WSJ NOCASE	92.87	81.92	74.31
BSC	93.71	84.32	76.63
DIRECT-CLICK	93.50	84.46	77.48
WSJ + QTB NOCASE	93.08	82.70	74.65
BSC	93.57	83.90	77.27
DIRECT-CLICK	93.43	84.11	**78.15**

Table 2: Tagging accuracies for different baseline settings and two transfer methods.DIRECT-CLICK is the approach we propose (see text). Column MS-251 NVX evaluates with tags from Bendersky et al. (2010). Their baseline is 89.3% and they report 91.6% for their method. MS-251 and Long-tail use tags from Section 3.1. We observe snippets for 2/500 long-tail queries and 31/251 MS-251 queries.

ent in Table 2. Firstly, a large part of the benefit of transfer has to do with case information that is available in the snippets but is missing in the query. The uncased tagger is insensitive to this mismatch and achieves significantly better results than the cased taggers. However, transferring information from the snippets provides additional benefits, significantly improving even the uncased baseline taggers. This is consistent with the analysis in Barr et al. (2008). Finally, we see that the direct transfer method from Section 2 significantly outperforms the method described in Bendersky et al. (2010). Table 3 confirms this trend when focusing on proper nouns, which are particularly difficult to identify in queries.

We also manually examined a set of 40 queries with their associated snippets, for which our best DIRECT-CLICK system made mistakes. In 32 cases, the errors in the query tagging could be traced back to errors in the snippet tagging. A better snippet tagger could alleviate that problem. In the remaining 8 cases there were problems with the matching – either the mis-tagged word was not found at all, or it was matched incorrectly. For example one of the results for the query "bell helmet" had a snippet containing "Bell cycling helmets" and we failed to match helmet to helmets.

Method	P	R	F
WSJ + QTB NOCASE	72.12	**79.80**	75.77
BSC	82.87	69.05	75.33
BSC + URL	**83.01**	70.80	76.42
DIRECT-CLICK	79.57	76.51	**78.01**
DIRECT-ALL	75.88	78.38	77.11
DIRECT-TOP-1	78.38	76.40	77.38

Table 3: Precision and recall of the NNP tag on the long-tail data for the best baseline method and the three transfer methods using that baseline.

5 Related Work

Barr et al. (2008) manually annotate a corpus of 2722 queries with 19 POS tags and use it to train and evaluate POS taggers, and also describe the linguistic structures they find. Unfortunately their data is not available so we cannot use it to compare to their results. Rüd et al. (2011) create features based on search engine results, that they use in an NER system applied to queries. They report report significant improvements when incorporating features from the snippets. In particular, they exploit capitalization and query terms matching URL components; both of which we have used in this work. Li et al. (2009) use clicks in a product data base to train a tagger for product queries, but they do not use snippets and do not annotate syntax. Li (2010) and Manshadi and Li (2009) also work on adding tags to queries, but do not use snippets or search logs as a source of information.

6 Conclusions

We described a simple method for training a search-query POS tagger from search-logs by transferring context from relevant snippet sets to query terms. We compared our approach to previous work, achieving an error reduction of 20%. In contrast to the approach proposed by Bendersky et al. (2010), our approach does not require access to the search engine or index when tagging a new query. By explicitly re-training our final model, it has the ability to pool knowledge from several related queries and incorporate the information into the model parameters. An area for future work is to transfer other syntactic information, such as parse structures or supertags using a similar transfer approach.

References

Cory Barr, Rosie Jones, and Moira Regelson. 2008. The linguistic structure of English web-search queries. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1021–1030, Honolulu, Hawaii, October. Association for Computational Linguistics.

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 international conference on Information and knowledge management*, pages 1537–1540. ACM.

M. Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proc. of EMNLP*.

John Judge, Aoife Cahill, and Josef van Genabith. 2006. Questionbank: Creating a corpus of parse-annotated questions. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 497–504, Sydney, Australia, July. Association for Computational Linguistics.

X. Li, Y.Y. Wang, and A. Acero. 2009. Extracting structured information from user queries with semi-supervised conditional random fields. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 572–579. ACM.

X. Li. 2010. Understanding the semantic structure of noun phrase queries. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1337–1345. Association for Computational Linguistics.

M. Manshadi and X. Li. 2009. Semantic tagging of web search queries. 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*, pages 861–869. Association for Computational Linguistics.

M. P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn treebank. *Computational Linguistics*, 19.

S. Petrov, D. Das, and R. McDonald. 2012. A universal part-of-speech tagset. In *Proc. of LREC*.

Stefan Rüd, Massimiliano Ciaramita, Jens Müller, and Hinrich Schütze. 2011. Piggyback: Using search engines for robust cross-domain named entity recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 965–975, Portland, Oregon, USA, June. Association for Computational Linguistics.

J. Uszkoreit and T. Brants. 2008. Distributed word clustering for large scale class-based language modeling in machine translation. In *Proc. of ACL*.

Toward Automatically Assembling Hittite-Language Cuneiform Tablet Fragments into Larger Texts

Stephen Tyndall
University of Michigan
styndall@umich.edu

Abstract

This paper presents the problem within Hittite and Ancient Near Eastern studies of fragmented and damaged cuneiform texts, and proposes to use well-known text classification metrics, in combination with some facts about the structure of Hittite-language cuneiform texts, to help classify a number of fragments of clay cuneiform-script tablets into more complete texts. In particular, I propose using Sumerian and Akkadian ideogrammatic signs within Hittite texts to improve the performance of Naive Bayes and Maximum Entropy classifiers. The performance in some cases is improved, and in some cases very much not, suggesting that the variable frequency of occurrence of these ideograms in individual fragments makes considerable difference in the ideal choice for a classification method. Further, complexities of the writing system and the digital availability of Hittite texts complicate the problem.

1 Introduction

The Hittite empire, in existence for about 600 years between 1800 and 1200 BCE, left numerous historical, political, and literary documents behind, written in cuneiform in clay tablets. There are a number of common problems that confront Hittite scholars interested in any subdiscipline of Hittitology, be it history, philology, or linguistics. Horst Klengel summarizes the issue most crucial to this paper:

> Some general problems, affecting both philologists and historians, are caused by the Hittite textual tradition itself. First, the bulk of the cuneiform material is fragmentary. The tablets, discovered in various depots in the Hittite capital and in some provincial centers, normally were of a larger size. When the archives were destroyed, the tablets for the most part broke into many pieces. Therefore, the joining of fragments became an important prerequisite for interpretation(Klengel, 2002).

Most Hittite texts are broken, but a number exist in more than one fragmentary copy.

Figure 1 shows a photograph, taken from the University of Meinz *Konkordanz der hethitischen Texte*[1], of a typical Hittite cuneiform fragment.

Complete or partially-complete texts are assembled from collections of fragments based on shape, writing size and style, and sentence similarity. Joins between fragments are not made systematically, but are usually discovered by scholars assembling large numbers of fragments that reference a specific subject, like some joins recently made in Hittite treaty documents in (Beckman, 1997).

Joins are thus fairly rare compared to the frequency of new publishing of fragments. Such joins and the larger texts created therewith are catalogued according to a CTH (*Catalogue des Textes Hittites*[2]) number. Each individual text is composed of one or more cuneiform fragments belonging to one or more copies of a single original work.

[1] available at http://www.hethport.uni-wuerzburg.de/HPM/hethportlinks.html

[2] available at http://www.hethport.uni-wuerzburg.de/CTH/

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 243–247,
Jeju, Republic of Korea, 8-14 July 2012.

Figure 2 shows a published join in hand-copied cuneiform fragments. In this case, the fragments are not contiguous, and only the text on the two fragments was used to make the join.

The task then, for the purposes of this paper, is to connect unknown fragments of Hittite cuneiform tablets with larger texts. I'm viewing this as a text classification task, where larger, CTH-numbered texts are the categories, and small fragments are the bits of text to be assigned to these categories.

2 The Corpus of Hittite

Hittite cuneiform consists of a mix of syllabic writing for Hittite words and logographic writing, typically Sumerian ideograms, standing in for Hittite words. Most words are written out phonologically using syllabic signs, in structure mostly CV and VC, and a few CVC. Some common words are written with logograms from other Ancient Near Eastern languages, e.g. Hittite *antuhša-* 'man' is commonly written with the Sumerian-language logogram transcribed LÚ. Such writings are called Sumerograms or Akkadograms, depending on the language from which the ideogram is taken.

The extant corpus of Hittite consists of more than 30,000 clay tablets and fragments excavated at sites in Turkey, Syria, and Egypt (Hoffner and Melchert, 2008, 2-3). Many of these fragments are assigned to one of the 835 texts catalogued in the CTH.

3 Prior Work

A large number of prior studies on text classification have informed the progress of this study. Categorization of texts into genres is very well studied (Dewdney et al., 2001). Other related text classification studies have looked at classifying text by source, in contexts of speech, as in an attempt to classify some segments of speech into native and non-native speaker categories (Tomokiyo and Jones, 2001), and writing and authorship, as in the famous Federalist Papers study(Mosteller and Wallace, 1984), and context, as in a categorization of a set of articles according to which newspaper they appeared in (Argamon-Engelson et al., 1998).

Measures of similarity among sections of a single document bear a closer relation to this project than the works above. Previous studies have examined in-

Figure 1: Photograph of a Hittite Tablet Fragment

Figure 2: Published Fragment Join

ternal document similarity, using some vector-based metrics to judge whether documents maintain the same subject throughout (Nicholson, 2009).

Very little computational work on cuneiform languages or texts exists. The most notable example is a study that examined grapheme distribution as a way to understand Hurrian substratal interference in the orthography of Akkadian-language cuneiform texts written in the Hurrian-speaking town of Nuzi (Smith, 2007). Smith's work, though using different classifying methods and and an enormously different corpus on a language with different characteristics, is the most similar to this study, since both are attempts to classify cuneiform fragments into categories - in Smith's case, into Hurrian-influenced Nuzi Akkadian and non-Nuzi standard Akkadian.

4 The Project Corpus

For this project, I use a corpus of neo-Hittite fragment transcriptions available from H. Craig Melchert (Melchert,). The corpus is one large text file, divided into CTH numbered sections, which themselves are divided into fragments labeled by their publication numbers - mostly KUB, which stands for *Keilschrifturkunden aus Boghazköi* or KBo, *Keilschrifttexte aus Boghazköi*, the two major publications for Hittite text fragments.

I restricted the fragments used in this project to fragments belonging to texts known to exist in at least two copies, a choice that produces a larger number of fragments per text without requiring a judgment about what number of fragments in a text constitutes "fragmented enough" for a legitimate test of this task. This leaves 36 total CTH-numbered texts, consisting of 389 total fragments.

The fragments themselves are included as plain text, with restorations by the transcribers left intact and set off by brackets, in the manner typical of cuneiform transcription. In transcription, signs with phonemic value are written in lower case characters, while ideograms are represented in all caps. Sign boundaries are represented by a hyphen, indicating the next sign is part of the current word, by an equals sign, indicating the next sign is a clitic, or a space, indicating that the next sign is part of a new word.

```
{KUB XXXI 25; DS 29}
x
[              ]A-NA KUR URUHa[t-ti?
[            i]s-tar-ni=sum-m[i
[             ]x nu=kn ki-x[
[             ] KUR URUMi-iz-ri=y[a
[is-tar-ni]=sum-mi e-es-du [

[    ] nu=kn A-NA KUR URUMi-iz-ri[
[A-NA EGI]R UDmi is-tar-ni=su[m-mi
```

This fragment, KUB XXI25, is very small and broken on both sides. The areas between brackets are sections of the text broken off or effaced by erosion of tablet surface material. Any text present between brackets has been inferred from context and transcriber experience with usual phrasing in Hittite. In the last line, the sign *EGIR*, a Sumerian ideogram, which is split by a bracket, was partially effaced but still recognizable to the transcriber, and so is split by a bracket.

5 Methods

For this project, I used both Naive Bayes and Maximum Entropy classifiers as implemented by the MAchine Learning for LanguagE Toolkit, MALLET(McCallum, 2002).

Two copies of the corpus were prepared. In one, anything in brackets or partially remaining after brackets was removed, leaving only characters actually preserved on the fragment. This copy is called *Plain Cuneiform* in the results section. The other has all bracket characters removed, leaving all actual characters and all characters suggested by the transcribers. This corpus is called *Brackets Removed* in the results section. By removing the brackets but leaving the suggested characters, I hoped to use the transcribers' intuitions about Hittite texts to further improve the performance of both classifiers.

The corpora were tokenized in two ways:

1. The tokens were defined only by spaces, capturing all words in the corpus.

2. The tokens were defined as a series of capital letters and punctuation marks, capturing only the Sumerian and Akkadian ideograms in the text, i.e. the very common Sumerian ideogram *DINGER.MEŠ*, 'the gods'.

The training and tests were all performed using MALLET's standard algorithms, cross-validated,

Table 1: Results for Plain Corpus

Tokenization	Naive Bayes	Max Ent
All Tokens	.55	.61
Ideograms Only	.44	.51

Table 2: Results for Tests on Corpus with Brackets Removed

Tokenization	Naive Bayes	Max Ent
All Tokens	.64	.67
Ideograms Only	.49	.54

splitting the data randomly into ten parts, and using 9 parts of the data as a training set and 1 part of the data as a test set. This means that each set was tested ten times, with all of the data eventually being used as part of the testing phase.

6 Results and Discussion

Accuracy values from the classifiers using the Plain corpus, and from the corpus with the Brackets Removed, are presented in Tables 1 and 2, respectively. The measures are raw accuracy, the fraction of the test fragments that the methods categorized correctly.

The results for the Plain Corpus show that the Naive Bayes classifier was 55% accurate with all tokens, and 44% accurate with ideograms alone. The Maximum Entropy classifier was 61% accurate with all tokens, and 51% accurate with ideograms only.

Both classifiers performed better with the Brackets Removed corpus. The Naive Bayes classifier was accurate 64% of the time with all tokens and 49% of the time with ideograms only. The Maximum Entropy classifier was 67% accurate with all tokens, and 54% accurate with ideograms only.

The predicted increase in accuracy using ideograms was not upheld by the above tests. It may be the case that Sumerograms and Akkadograms are insufficiently frequent, particularly in smaller fragments, to allow for correct categorization. Some early tests suggested occasional excellent results for this tokenization scheme, including a single random 90-10 training/test run that showed a test accuracy of .86, much higher than any larger cross-validated test included above. This suggests, perhaps unsurprisingly, that the accuracy of classification using Sumerograms and Akkadograms is heavily dependent on the structure of the fragments in question.

Maximum Entropy classification proved to be slightly better, in every instance, than Naive Bayes classification, a fact that will prove useful in future tests and applications.

The fact that removing the brackets and including the transcribers' additions improved the performance of all classifiers will likewise prove useful, since transcriptions of fragments are typically published with such bracketed additions. It also seems to demonstrate the quality of these additions made by transcribers.

Overall, these tests suggest that in general, the 'use-everything' approach is better for accurate classification of Hittite tablet fragments with larger CTH texts. However, in some cases, when the fragments in question have a large number of Sumerograms and Akkadograms, using them exclusively may be the right choice.

7 Implications and Further Work

In the future, I hope to continue with a number of other approaches to this problem, including lemmatizing the various Hittite noun and verb paradigms. Additionally, viewing the problem in other ways, e.g. regarding tablet fragments as elements for connection by clustering algorithms, might work well.

Given the large number of small fragments now coming to light, this method could speed the process of text assembly considerably. A new set of archives, recently discovered in the Hittite city of Šapinuwa, are only now beginning to see publication. This site contains more than 3000 new Hittite tablet fragments, with excavations ongoing(Süel, 2002). The jumbled nature of the dig site means that the process of assembling new texts from this site will be one of the major tasks in for Hittite scholars in the near future. This attempt at speeding the task is only the beginning of what I hope will be a considerable body of work to help build more complete texts, and therefore more complete literatures and histories, of not only Hittite, but other cuneiform languages like Akkadian and Sumerian, some of the world's earliest written languages.

References

S. Argamon-Engelson, M. Koppel, and G. Avneri. 1998. Style-based text categorization: What newspaper am i reading. In *Proc. of the AAAI Workshop on Text Categorization*, pages 1–4.

G. Beckman. 1997. New Joins to Hittite Treaties. *Zeitschrift für Assyriologie und Vorderasiatische Archäologie*, 87(1):96–100.

N. Dewdney, C. VanEss-Dykema, and R. MacMillan. 2001. The form is the substance: Classification of genres in text. In *Proceedings of the workshop on Human Language Technology and Knowledge Management-Volume 2001*, pages 1–8. Association for Computational Linguistics.

H.A. Hoffner and H.C. Melchert. 2008. *A grammar of the Hittite language*. Eisenbrauns.

Horst Klengel. 2002. Problems in hittite history, solved and unsolved. In Simrit Dhesi K. Aslihan Yener, Harry A. Hoffner Jr., editor, *Recent developments in Hittite archaeology and history: papers in memory of Hans G. Güterbock*, pages 101–109. Eisenbrauns.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

H. Craig Melchert. Anatolian databases. `http://www.linguistics.ucla.edu/people/Melchert/webpage/AnatolianDatabases.htm`.

F. Mosteller and D.L. Wallace. 1984. Applied bayesian and classical inference: The case of the federalist papers.

C. Nicholson. 2009. Judging whether a document changes in subject. In *Southeastcon, 2009. SOUTHEASTCON'09. IEEE*, pages 189–194. IEEE.

S.P. Smith. 2007. *Hurrian Orthographic Interference in Nuzi Akkadian: A Computational Comparative Graphemic Analysis*. Ph.D. thesis, Harvard University Cambridge, Massachusetts.

A. Süel. 2002. Ortaköy-sapinuwa. In Simrit Dhesi K. Aslihan Yener, Harry A. Hoffner Jr., editor, *Recent developments in Hittite archaeology and history: papers in memory of Hans G. Güterbock*, pages 157–165. Eisenbrauns.

L.M. Tomokiyo and R. Jones. 2001. You're not from 'round here, are you?: naive bayes detection of non-native utterance text. In *Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001*, pages 1–8. Association for Computational Linguistics.

A Corpus of Textual Revisions in Second Language Writing

John Lee and Jonathan Webster
The Halliday Centre for Intelligent Applications of Language Studies
Department of Chinese, Translation and Linguistics
City University of Hong Kong
{jsylee,ctjjw}@cityu.edu.hk

Abstract

This paper describes the creation of the first large-scale corpus containing drafts and final versions of essays written by non-native speakers, with the sentences aligned across different versions. Furthermore, the sentences in the drafts are annotated with comments from teachers. The corpus is intended to support research on textual revision by language learners, and how it is influenced by feedback. This corpus has been converted into an XML format conforming to the standards of the Text Encoding Initiative (TEI).

1 Introduction

Learner corpora have been playing an increasingly important role in both Second Language Acquisition and Foreign Language Teaching research (Granger, 2004; Nesi et al., 2004). These corpora contain texts written by non-native speakers of the language (Granger et al., 2009); many also annotate text segments where there are errors, and the corresponding error categories (Nagata et al., 2011). In addition, some learner corpora contain pairs of sentences: a sentence written by a learner of English as a second language (ESL), paired with its correct version produced by a native speaker (Dahlmeier and Ng, 2011). These datasets are intended to support the training of automatic text correction systems (Dale and Kilgarriff, 2011).

Less attention has been paid to how a language learner produces a text. Writing is often an iterative and interactive process, with cycles of textual revision, guided by comments from language teachers.

Discipline	# drafts
Applied Physics	988
Asian and International Studies	410
Biology	2310
Building Science and Technology	705
Business	1754
Computer Science	466
Creative Media	118
Electronic Engineering	1532
General Education	651
Law	31
Linguistics	2165
Management Sciences	1278
Social Studies	912
Total	13320

Table 1: Draft essays are collected from courses in various disciplines at City University of Hong Kong. These drafts include lab reports, data analysis, argumentative essays, and article summaries. There are 3760 distinct essays, most of which consist of two to four successive drafts. Each draft has on average 44.2 sentences, and the average length of a sentence is 13.3 words. In total, the corpus contains 7.9 million words.

Understanding the dynamics of this process would benefit not only language teachers, but also the design of writing assistance tools that provide automatic feedback (Burstein and Chodorow, 2004).

This paper presents the first large-scale corpus that will enable research in this direction. After a review of previous work (§2), we describe the design and a preliminary analysis of our corpus (§3).

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 248–252,
Jeju, Republic of Korea, 8-14 July 2012.

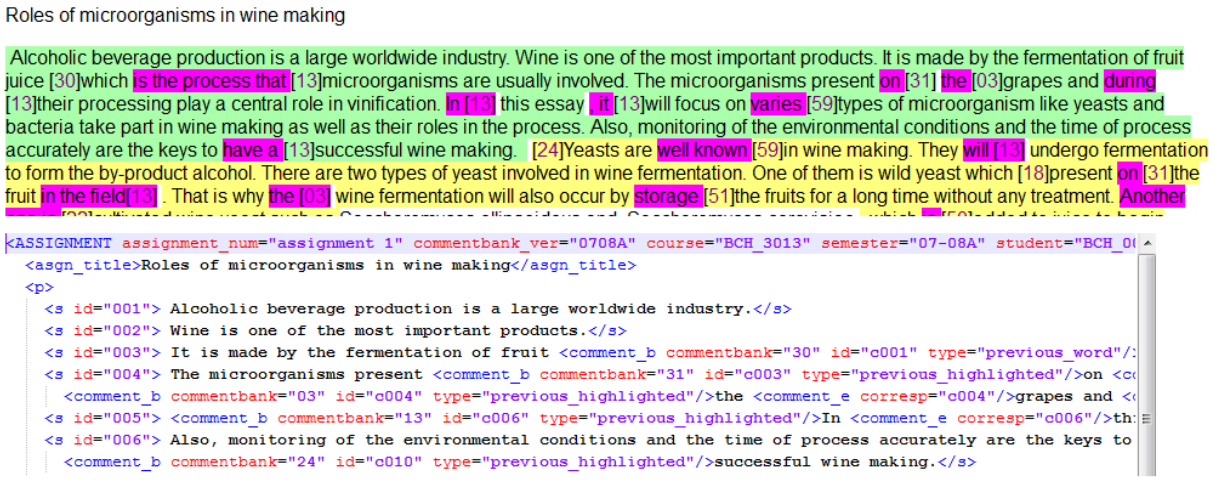


Figure 1: On top is a typical draft essay, interleaved with comments from a tutor (§3.2): two-digit codes from the Comment Bank are enclosed in angled brackets, while open-ended comments are enclosed in angled brackets. On the bottom is the same essay in TEI format, the output of the process described in §3.3.

2 Previous Research

In this section, we summarize previous research on feedback in language teaching, and on the nature of the revision process by language learners.

2.1 Feedback in Language Learning

Receiving feedback is a crucial element in language learning. While most agree that both the *form* and *content* of feedback plays an important role, there is no consensus on their effects. Regarding form, some argue that direct feedback (providing corrections) are more effective in improving the quality of writing than indirect feedback (pointing out an error but not providing corrections) (Sugita, 2006), but others reached opposite conclusions (Ferris, 2006; Lee, 2008).

Regarding content, it has been observed that teachers spend a disproportionate amount of time on identifying word-level errors, at the expense of those at higher levels, such as coherence (Furneaux et al., 2007; Zamel, 1985). There has been no large-scale empirical study, however, on the effectiveness of feedback at the paragraph or discourse levels.

2.2 Revision Process

While text editing in general has been analyzed (Mahlow and Piotrowski, 2008), the nature of revisions by language learners — for example, whether learners mostly focus on correcting mechanical, word-level errors, or also substantially reorganize paragraph or essay structures — has hardly been investigated. One reason for this gap in the literature is the lack of corpus data: none of the existing learner corpora (Izumi et al., 2004; Granger et al., 2009; Nagata et al., 2011; Dahlmeier and Ng, 2011) contains drafts written by non-native speakers that led to the "final version". Recently, two corpora with text revision information have been compiled (Xue and Hwa, 2010; Mizumoto et al., 2011), but neither contain feedback from language teachers. Our corpus will allow researchers to not only examine the revision process, but also investigate any correlation with the amount and type of feedback.

3 Corpus Description

We first introduce the context in which our data was collected (§3.1), then describe the kinds of comments in the drafts (§3.2). We then outline the conversion process of the corpus into XML format (§3.3), followed by an evaluation (§3.4) and an analysis (§3.5).

3.1 Background

Between 2007 and 2010, City University of Hong Kong hosted a language learning project where English-language tutors reviewed and provided feedback on academic essays written by students,

Paragraph level		Sentence level		Word level	
Coherence: more elaboration is needed	680	Conjunction missing	1554	Article missing	10586
Paragraph: new paragraph	522	Sentence: new sentence	1389	Delete this	9224
Coherence: sign posting	322	Conjunction: wrong use	923	Noun: countable	7316
Coherence: missing topic sentence	222	Sentence: fragment	775	Subject-verb agreement	4008

Table 2: The most frequent error categories from the Comment Bank, aimed at errors at different levels.

most of whom were native speakers of Chinese (Webster et al., 2011). More than 300 TESOL students served as language tutors, and over 4,200 students from a wide range of disciplines (see Table 1) took part in the project.

For each essay, a student posted a first draft[1] as a blog on an e-learning environment called Blackboard Academic Suite; a language tutor then directly added comments on the blog. Figure 1 shows an example of such a draft. The student then revised his or her draft and may re-post it to receive further comments. Most essays underwent two revision cycles before the student submitted the final version.

3.2 Comments

Comments in the draft can take one of three forms:

Code The tutor may insert a two-digit code, representing one of the 60 common error categories in our "Comment Bank", adopted from the XWiLL project (Wible et al., 2001). These categories address issues ranging from the word level to paragraph level (see Table 2), with a mix of direct (e.g., "new paragraph") and indirect feedback (e.g., "more elaboration is needed").

Open-ended comment The tutor may also provide personally tailored comments.

Hybrid Both a code and an open-ended comment.

For every comment[2], the tutor highlights the problematic words or sentences at which it is aimed. Sometimes, general comments about the draft as a whole are also inserted at the beginning or the end.

[1] In the rest of the paper, these drafts will be referred to "version 1", "version 2", and so on.

[2] Except those comments indicating that a word is missing.

3.3 Conversion to XML Format

The data format for the essays and comments was not originally conceived for computational analysis. The drafts, downloaded from the blog entries, are in HTML format, with comments interspersed in them; the final versions are Microsoft Word documents. Our first task, therefore, is to convert them into a machine-actionable, XML format conforming to the standards of the Text Encoding Initiative (TEI). This conversion consists of the following steps:

Comment extraction After repairing irregularities in the HTML tags, we eliminated attributes that are irrelevant to comment extraction, such as font and style. We then identified the Comment Bank codes and open-ended comments.

Comment-to-text alignment Each comment is aimed at a particular text segment. The text segment is usually indicated by highlighting the relevant words or changing their background color. After consolidating the tags for highlighting and colors, our algorithm looks for the nearest, preceding text segment with a color different from that of the comment.

Title and metadata extraction From the top of the essay, our algorithm scans for short lines with metadata such as the student and tutor IDs, semester and course codes, and assignment and version numbers. The first sentence in the essay proper is taken to be the title.

Sentence segmentation Off-the-shelf sentence segmentators tend to be trained on newswire texts (Reynar and Ratnaparkhi, 1997), which significantly differ from the noisy text in our corpus. We found it adequate to use a stop-list, supplemented with a few regular expressions

Evaluation	Precision	Recall
Comment extraction		
- *code*	94.7%	100%
- *open-ended*	61.8%	78.3%
Comment-to-text alignment	86.0%	85.2%
Sentence segmentation	94.8%	91.3%

Table 3: Evaluation results of the conversion process described in §3.3. Precision and recall are calculated on correct detection of the start and end points of comments and boundaries.

that detect exceptions, such as abbreviations and digits.

Sentence alignment Sentences in consecutive versions of an essay are aligned using cosine similarity score. To allow dynamic programming, alignments are limited to one-to-one, one-to-two, two-to-one, or two-to-two[3]. Below a certain threshold[4], a sentence is no longer aligned, but is rather considered inserted or deleted. The alignment results are stored in the XCES format (Ide et al., 2002).

3.4 Conversion Evaluation

To evaluate the performance of the conversion algorithm described in §3.3, we asked a human to manually construct the TEI XML files for 14 pairs of draft versions. These gold files are then compared to the output of our algorithm. The results are shown in Table 3.

In comment extraction, codes can be reliably identified. Among the open-ended comments, however, those at the beginning and end of the drafts severely affected the precision, since they are often not quoted in brackets and are therefore indistinguishable from the text proper. In comment-to-text alignment, most errors were caused by inconsistent or missing highlighting and background colors.

The accuracy of sentence alignment is 89.8%, measured from the perspective of sentences in Version 1. It is sometimes difficult to decide whether a sentence has simply been edited (and should therefore be aligned), or has been deleted with a new sentence inserted in the next draft.

[3]That is, the order of two sentences is flipped.

[4]Tuned to 0.5 based on a random subset of sentence pairs.

3.5 Preliminary Analysis

As shown in Table 4, the tutors were much more likely to use codes than to provide open-ended comments. Among the codes, they overwhelmingly emphasized word-level issues, echoing previous findings (§2.1). Table 2 lists the most frequent codes. Missing articles, noun number and subject-verb agreement round out the top errors at the word level, similar to the trend for Japanese speakers (Lee and Seneff, 2008). At the sentence level, conjunctions turn out to be challenging; at the paragraph level, paragraph organization, sign posting, and topic sentence receive the most comments.

In a first attempt to gauge the utility of the comments, we measured their density across versions. Among Version 1 drafts, a code appears on average every 40.8 words, while an open-ended comment appears every 84.7 words. The respective figures for Version 2 drafts are 65.9 words and 105.0 words. The lowered densities suggest that students were able to improve the quality of their writing after receiving feedback.

Comment Form	Frequency
Open-ended	47072
Hybrid	1993
Code	88370
- *Paragraph level*	*3.2%*
- *Sentence level*	*6.0%*
- *Word level*	*90.8%*

Table 4: Distribution of the three kinds of comments (§3.2), with the Comment Bank codes further subdivided into different levels (See Table 2).

4 Conclusion and Future Work

We have presented the first large-scale learner corpus which contains not only texts written by non-native speakers, but also the successive drafts leading to the final essay, as well as teachers' comments on the drafts. The corpus has been converted into an XML format conforming to TEI standards.

We plan to port the corpus to a platform for text visualization and search, and release it to the research community. It is expected to support studies on textual revision of language learners, and the effects of different types of feedback.

Acknowledgments

We thank Shun-shing Tsang for his assistance with implementing the conversion and performing the evaluation. This project was partially funded by a Strategic Research Grant (#7008065) from City University of Hong Kong.

References

Jill Burstein and Martin Chodorow. 2004. Automated Essay Evaluation: The Criterion online writing service. *AI Magazine*.

Daniel Dahlmeier and Hwee Tou Ng. 2011. Grammatical Error Correction with Alternating Structure Optimization. *Proc. ACL.*

Robert Dale and Adam Kilgarriff. 2011. Helping Our Own: The HOO 2011 Pilot Shared Task. *Proc. European Workshop on Natural Language Generation (ENLG)*, Nancy, France.

Dana Ferris. 2006. Does Error Feedback Help Student Writers? New Evidence on the Short- and Long-Term Effects of Written Error Correction. In *Feedback in Second Language Writing: Contexts and Issues*, Ken Hyland and Fiona Hyland (eds). Cambridge University Press.

Clare Furneaux, Amos Paran, and Beverly Fairfax. 2007. Teacher Stance as Reflected in Feedback on Student Writing: An Empirical Study of Secondary School Teachers in Five Countries. *International Review of Applied Linguistics in Language Teaching* 45(1): 69-94.

Sylviane Granger. 2004. Computer Learner Corpus Research: Current Status and Future Prospect. *Language and Computers* 23:123–145.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. 2009. International Corpus of Learner English v2. Presses universitaires de Louvain, Belgium.

Nancy Ide, Patrice Bonhomme, and Laurent Romary. 2000. XCES: An XML-based Encoding Standard for Linguistic Corpora. *Proc. LREC*.

Emi Izumi, Kiyotaka Uchimoto, and Hitoshi Isahara. 2004. The NICT JLE Corpus: Exploiting the Language Learners' Speech Database for Research and Education. *International Journal of the Computer, the Internet and Management* 12(2):119–125.

Icy Lee. 2008. Student Reactions to Teacher Feedback in Two Hong Kong Secondary Classrooms. *Journal of Second Language Writing* 17(3):144-164.

John Lee and Stephanie Seneff. 2008. An Analysis of Grammatical Errors in Nonnative Speech in English. *Proc. IEEE Workshop on Spoken Language Technology.*

Cerstin Mahlow and Michael Piotrowski. 2008. Linguistic Support for Revising and Editing. *Proc. International Conference on Computational Linguistics and Intelligent Text Processing*.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining Revision Log of Language Learning SNS for Automated Japanese Error Correction of Second Language Learners. *Proc. IJCNLP*.

Ryo Nagata, Edward Whittaker, and Vera Sheinman. 2011. Creating a Manually Error-tagged and Shallow-parsed Learner Corpus. *Proc. ACL.*

Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A Maximum Entropy Approach to Identifying Sentence Boundaries. *Proc. 5th Conference on Applied Natural Language Processing*, Washington DC.

Yoshihito Sugita. 2006. The Impact of Teachers' Comment Types on Students' Revision. *ELT Journal* 60(1):34–41.

Hilary Nesi, Gerard Sharpling, and Lisa Ganobcsik-Williams. 2004. Student Papers Across the Curriculum: Designing and Developing a Corpus of British Student Writing. *Computers and Composition* 21(4):439–450.

Frank Tuzi. 2004. The Impact of E-Feedback on the Revisions of L2 Writers in an Academic Writing Course. *Computers and Composition* 21(2):217-235.

Jonathan Webster, Angela Chan, and John Lee. 2011. Online Language Learning for Addressing Hong Kong Tertiary Students' Needs in Academic Writing. *Asia Pacific World* 2(2):44–65.

David Wible, Chin-Hwa Kuo, Feng-Li Chien, Anne Liu, and Nai-Lung Tsao. 2001. A Web-Based EFL Writing Environment: Integrating Information for Learners, Teachers, and Researchers. *Computers and Education* 37(34):297-315.

Huichao Xue and Rebecca Hwa. 2010. Syntax-Driven Machine Translation as a Model of ESL Revision. *Proc. COLING*.

Vivian Zamel. 1985. Responding to Student Writing. *TESOL Quarterly* 19(1):79-101.

Coarse Lexical Semantic Annotation with Supersenses: An Arabic Case Study

Nathan Schneider[†] **Behrang Mohit**[*] **Kemal Oflazer**[*] **Noah A. Smith**[†]
School of Computer Science, Carnegie Mellon University
[*]Doha, Qatar [†]Pittsburgh, PA 15213, USA
{nschneid@cs.,behrang@,ko@cs.,nasmith@cs.}cmu.edu

Abstract

"Lightweight" semantic annotation of text calls for a simple representation, ideally without requiring a semantic lexicon to achieve good coverage in the language and domain. In this paper, we repurpose WordNet's *supersense tags* for annotation, developing specific guidelines for nominal expressions and applying them to Arabic Wikipedia articles in four topical domains. The resulting corpus has high coverage and was completed quickly with reasonable inter-annotator agreement.

1 Introduction

The goal of "lightweight" semantic annotation of text, particularly in scenarios with limited resources and expertise, presents several requirements for a representation: simplicity; adaptability to new languages, topics, and genres; and coverage. This paper describes coarse lexical semantic annotation of Arabic Wikipedia articles subject to these constraints. Traditional lexical semantic representations are either narrow in scope, like *named entities*,[1] or make reference to a full-fledged *lexicon/ontology*, which may insufficiently cover the language/domain of interest or require prohibitive expertise and effort to apply.[2] We therefore turn to **supersense tags** (SSTs), 40 coarse lexical semantic classes (25 for nouns, 15 for verbs) originating in WordNet. Previously these served as groupings of English lexicon entries, but here we have repurposed them as target labels for direct human annotation.

أن القياسية للأرقام جينيس كتاب يعتبر
considers book Guinness for-records the-standard that
COMMUNICATION

في جامعة أقدم المغرب فاس في القيروان جامعة
university Al-Karaouine in Fez Morocco oldest university in
ARTIFACT LOCATION GROUP

العالم تم حيث تأسيسها في 859 سنة ميلادي .
the-world where was established in year AD
LOCATION ACT TIME

'*The Guinness Book of World Records* considers the University of Al-Karaouine in Fez, Morocco, established in the year 859 AD, the oldest university in the world.'

Figure 1: A sentence from the article "Islamic Golden Age," with the supersense tagging from one of two annotators. The Arabic is shown left-to-right.

Part of the earliest versions of WordNet, the supersense categories (originally, "lexicographer classes") were intended to partition *all* English noun and verb senses into broad groupings, or *semantic fields* (Miller, 1990; Fellbaum, 1990). More recently, the task of automatic supersense tagging has emerged for English (Ciaramita and Johnson, 2003; Curran, 2005; Ciaramita and Altun, 2006; Paaß and Reichartz, 2009), as well as for Italian (Picca et al., 2008; Picca et al., 2009; Attardi et al., 2010) and Chinese (Qiu et al., 2011), languages with WordNets mapped to English WordNet.[3] In principle, we believe supersenses ought to apply to nouns and verbs in *any* language, and need not depend on the availability of a semantic lexicon.[4] In this work we focus on the noun SSTs, summarized in figure 2 and applied to an Arabic sentence in figure 1.

SSTs both refine and relate lexical items: they capture lexical polysemy on the one hand—e.g.,

[1]Some ontologies like those in Sekine et al. (2002) and BBN Identifinder (Bikel et al., 1999) include a large selection of classes, which tend to be especially relevant to proper names.

[2]E.g., a WordNet (Fellbaum, 1998) sense annotation effort reported by Passonneau et al. (2010) found considerable inter-annotator variability for some lexemes; FrameNet (Baker et al., 1998) is limited in coverage, even for English; and PropBank (Kingsbury and Palmer, 2002) does not capture semantic relationships across lexemes. We note that the Omega ontology (Philpot et al., 2003) has been used for fine-grained cross-lingual annotation (Hovy et al., 2006; Dorr et al., 2010).

[3]Note that work in supersense tagging used text with *fine-grained* sense annotations that were then coarsened to SSTs.

[4]The noun/verb distinction might prove problematic in some languages.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 253–258,
Jeju, Republic of Korea, 8-14 July 2012.

Crusades · Damascus · Ibn Tolun Mosque · Imam Hussein Shrine · Islamic Golden Age · Islamic History · Ummayad Mosque	434s	16,185t	5,859m
Atom · Enrico Fermi · Light · Nuclear power · Periodic Table · Physics · Muhammad al-Razi	777s	18,559t	6,477m
2004 Summer Olympics · Christiano Ronaldo · Football · FIFA World Cup · Portugal football team · Raúl Gonzáles · Real Madrid	390s	13,716t	5,149m
Computer · Computer Software · Internet · Linux · Richard Stallman · Solaris · X Window System	618s	16,992t	5,754m

Table 1: Snapshot of the supersense-annotated data. The 7 article titles (translated) in each domain, with total counts of sentences, tokens, and supersense mentions. Overall, there are 2,219 sentences with 65,452 tokens and 23,239 mentions (1.3 tokens/mention on average). Counts exclude sentences marked as problematic and mentions marked ?.

disambiguating PERSON vs. POSSESSION for the noun **principal**—and generalize across lexemes on the other—e.g., **principal**, **teacher**, and **student** can all be PERSONs. This lumping property might be expected to give too much latitude to annotators; yet we find that in practice, it is possible to elicit reasonable inter-annotator agreement, even for a language other than English. We encapsulate our interpretation of the tags in a set of brief guidelines that aims to be usable by anyone who can read and understand a text in the target language; our annotators had no prior expertise in linguistics or linguistic annotation.

Finally, we note that ad hoc categorization schemes not unlike SSTs have been developed for purposes ranging from question answering (Li and Roth, 2002) to animacy hierarchy representation for corpus linguistics (Zaenen et al., 2004). We believe the interpretation of the SSTs adopted here can serve as a single starting point for diverse resource engineering efforts and applications, especially when fine-grained sense annotation is not feasible.

2 Tagging Conventions

WordNet's definitions of the supersenses are terse, and we could find little explicit discussion of the specific rationales behind each category. Thus, we have crafted more specific explanations, summarized for nouns in figure 2. English examples are given, but the guidelines are intended to be language-neutral. A more systematic breakdown, formulated as a 43-rule decision list, is included with the corpus.[5] In developing these guidelines we consulted English WordNet (Fellbaum, 1998) and SemCor (Miller et al., 1993) for examples and synset definitions, occasionally making simplifying decisions where we found distinctions that seemed esoteric or internally inconsistent. Special cases (e.g., multiword expressions, anaphora, figurative language) are addressed with additional rules.

3 Arabic Wikipedia Annotation

The annotation in this work was on top of a small corpus of Arabic Wikipedia articles that had already been annotated for named entities (Mohit et al., 2012). Here we use two different annotators, both native speakers of Arabic attending a university with English as the language of instruction.

Data & procedure. The dataset (table 1) consists of the main text of 28 articles selected from the topical domains of history, sports, science, and technology. The annotation task was to identify and categorize **mentions**, i.e., occurrences of terms belonging to noun supersenses. Working in a custom, browser-based interface, annotators were to tag each relevant token with a supersense category by selecting the token and typing a tag symbol. Any token could be marked as continuing a multiword unit by typing <. If the annotator was ambivalent about a token they were to mark it with the ? symbol. Sentences were pre-tagged with suggestions where possible.[6] Annotators noted obvious errors in sentence splitting and grammar so ill-formed sentences could be excluded.

Training. Over several months, annotators alternately annotated sentences from 2 designated articles of each domain, and reviewed the annotations for consistency. All tagging conventions were developed collaboratively by the author(s) and annotators during this period, informed by points of confusion and disagreement. WordNet and SemCor were consulted as part of developing the guidelines, but not during annotation itself so as to avoid complicating the annotation process or overfitting to WordNet's idiosyncracies. The training phase ended once inter-annotator mention F_1 had reached 75%.

[5] For example, one rule states that all man-made structures (buildings, rooms, bridges, etc.) are to be tagged as ARTIFACTs.

[6] Suggestions came from the previous named entity annotation of PERSONs, organizations (GROUP), and LOCATIONs, as well as heuristic lookup in lexical resources—Arabic WordNet entries (Elkateb et al., 2006) mapped to English WordNet, and named entities in OntoNotes (Hovy et al., 2006).

O **NATURAL OBJECT** natural feature or nonliving object in nature **barrier_reef nest neutron_star planet sky fishpond metamorphic_rock Mediterranean cave stepping_stone boulder Orion ember universe**

A **ARTIFACT** man-made structures and objects **bridge restaurant bedroom stage cabinet toaster antidote aspirin**

L **LOCATION** any name of a geopolitical entity, as well as other nouns functioning as locations or regions **Cote_d'Ivoire New_York_City downtown stage_left India Newark interior airspace**

P **PERSON** humans or personified beings; names of social groups (ethnic, political, etc.) that can refer to an individual in the singular **Persian_deity glasscutter mother kibbutznik firstborn worshiper Roosevelt Arab consumer appellant guardsman Muslim American communist**

G **GROUP** groupings of people or objects, including: organizations/institutions; followers of social movements **collection flock army meeting clergy Mennonite_Church trumpet_section health_profession peasantry People's_Party U.S._State_Department University_of_California population consulting_firm communism Islam** (= set of Muslims)

$ **SUBSTANCE** a material or substance **krypton mocha atom hydrochloric_acid aluminum sand cardboard DNA**

H **POSSESSION** term for an entity involved in ownership or payment **birthday_present tax_shelter money loan**

T **TIME** a temporal point, period, amount, or measurement **10_seconds day Eastern_Time leap_year 2nd_millenium_BC 2011** (= year) **velocity frequency runtime latency/delay middle_age half_life basketball_season words_per_minute curfew industrial_revolution instant/moment August**

= **RELATION** relations between entities or quantities **ratio scale reverse personal_relation exponential_function angular_position unconnectedness transitivity**

Q **QUANTITY** quantities and units of measure, including cardinal numbers and fractional amounts **7_cm 1.8_million 12_percent/12% volume** (= spatial extent) **volt real_number square_root digit 90_degrees handful ounce half**

F **FEELING** subjective emotions **indifference wonder murderousness grudge desperation astonishment suffering**

M **MOTIVE** an abstract external force that causes someone to intend to do something **reason incentive**

C **COMMUNICATION** information encoding and transmission, except in the sense of a physical object **grave_accent Book_of_Common_Prayer alphabet Cree_language onomatopoeia reference concert hotel_bill broadcast television_program discussion contract proposal equation denial sarcasm concerto software**

^ **COGNITION** aspects of mind/thought/knowledge/belief/perception; techniques and abilities; fields of academic study; social or philosophical movements referring to the system of beliefs **Platonism hypothesis logic biomedical_science necromancy hierarchical_structure democracy innovativeness vocational_program woodcraft reference visual_image Islam** (= Islamic belief system) **dream scientific_method consciousness puzzlement skepticism reasoning design intuition inspiration muscle_memory skill aptitude/talent method sense_of_touch awareness**

S **STATE** stable states of affairs; diseases and their symptoms **symptom reprieve potency poverty altitude_sickness tumor fever measles bankruptcy infamy opulence hunger opportunity darkness** (= lack of light)

@ **ATTRIBUTE** characteristics of people/objects that can be judged **resilience buxomness virtue immateriality admissibility coincidence valence sophistication simplicity temperature** (= degree of hotness) **darkness** (= dark coloring)

! **ACT** things people do or cause to happen; learned professions **meddling malpractice faith_healing dismount carnival football_game acquisition engineering** (= profession)

E **EVENT** things that happens at a given place and time **bomb_blast ordeal miracle upheaval accident tide**

R **PROCESS** a sustained phenomenon or one marked by gradual changes through a series of states **oscillation distillation overheating aging accretion/growth extinction evaporation**

X **PHENOMENON** a physical force or something that happens/occurs **electricity suction tailwind tornado effect**

+ **SHAPE** two and three dimensional shapes

D **FOOD** things used as food or drink

B **BODY** human body parts, excluding diseases and their symptoms

Y **PLANT** a plant or fungus

N **ANIMAL** non-human, non-plant life

Science chemicals, molecules, atoms, and subatomic particles are tagged as SUBSTANCE

Sports championships/tournaments are EVENTs

(Information) Technology Software names, kinds, and components are tagged as COMMUNICATION (e.g. **kernel, version, distribution, environment**). A **connection** is a RELATION; **project, support**, and a **configuration** are tagged as COGNITION; **development** and **collaboration** are ACTs.

Arabic conventions *Masdar* constructions (verbal nouns) are treated as nouns. Anaphora are not tagged.

Figure 2: *Above:* The complete supersense tagset for nouns; each tag is briefly described by its `symbol`, NAME, short description, and **examples**. Some examples and longer descriptions have been omitted due to space constraints. *Below:* A few domain- and language-specific elaborations of the general guidelines.

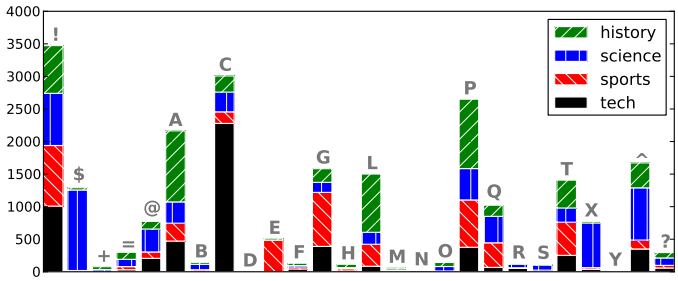


Figure 3: Distribution of supersense mentions by domain (left), and counts for tags occurring over 800 times (below). (Counts are of the *union* of the annotators' choices, even when they disagree.)

tag	*num*	*tag*	*num*
ACT (!)	3473	LOCATION (G)	1583
COMMUNICATION (C)	3007	GROUP (L)	1501
PERSON (P)	2650	TIME (T)	1407
ARTIFACT (A)	2164	SUBSTANCE ($)	1291
COGNITION (^)	1672	QUANTITY (Q)	1022

Main annotation. After training, the two annotators proceeded on a per-document basis: first they worked together to annotate several sentences from the beginning of the article, then each was independently assigned about half of the remaining sentences (typically with 5–10 shared to measure agreement). Throughout the process, annotators were encouraged to discuss points of confusion with each other, but each sentence was annotated in its entirety and never revisited. Annotation of 28 articles required approximately 100 annotator-hours. Articles used in pilot rounds were re-annotated from scratch.

Analysis. Figure 3 shows the distribution of SSTs in the corpus. Some of the most concrete tags—BODY, ANIMAL, PLANT, NATURAL OBJECT, and FOOD—were barely present, but would likely be frequent in life sciences domains. Others, such as MOTIVE, POSSESSION, and SHAPE, are limited in scope.

To measure inter-annotator agreement, 87 sentences (2,774 tokens) distributed across 19 of the articles (not including those used in pilot rounds) were annotated independently by each annotator. Inter-annotator mention F_1 (counting agreement over entire mentions and their labels) was 70%. Excluding the 1,397 tokens left blank by both annotators, the token-level agreement rate was 71%, with Cohen's $\kappa = 0.69$, and token-level F_1 was 83%.[7]

We also measured agreement on a tag-by-tag basis. For 8 of the 10 most frequent SSTs (figure 3), inter-annotator mention F_1 ranged from 73% to 80%. The two exceptions were QUANTITY at 63%, and COGNITION (probably the most heterogeneous category) at 49%. An examination of the confusion matrix reveals four pairs of supersense categories that tended to provoke the most disagreement: COMMUNICATION/COGNITION, ACT/COGNITION, ACT/PROCESS, and ARTIFACT/COMMUNICATION. The last is exhibited for the first mention in figure 1, where one annotator chose ARTIFACT (referring to the *physical* book) while the other chose COMMUNICATION (the *content*). Also in that sentence, annotators disagreed on the second use of *university* (ARTIFACT vs. GROUP). As with any sense annotation effort, some disagreements due to legitimate ambiguity and different interpretations of the tags—especially the broadest ones—are unavoidable.

A "soft" agreement measure (counting as matches any two mentions with the same label and at least one token in common) gives an F_1 of 79%, showing that boundary decisions account for a major portion of the disagreement. E.g., the city *Fez, Morocco* (figure 1) was tagged as a single LOCATION by one annotator and as two by the other. Further examples include the technical term 'thin client', for which one annotator omitted the adjective; and 'World Cup Football Championship', where one annotator tagged the entire phrase as an EVENT while the other tagged 'football' as a separate ACT.

[7] Token-level measures consider both the supersense label and whether it begins or continues the mention.

4 Conclusion

We have codified supersense tags as a simple annotation scheme for coarse lexical semantics, and have shown that supersense annotation of Arabic Wikipedia can be rapid, reliable, and robust (about half the tokens in our data are covered by a nominal supersense). Our tagging guidelines and corpus are available for download at `http://www.ark.cs.cmu.edu/ArabicSST/`.

Acknowledgments

We thank Nourhen Feki and Sarah Mustafa for assistance with annotation, as well as Emad Mohamed, CMU ARK members, and anonymous reviewers for their comments. This publication was made possible by grant NPRP-08-485-1-083 from the Qatar National Research Fund (a member of the Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

Giuseppe Attardi, Stefano Dei Rossi, Giulia Di Pietro, Alessandro Lenci, Simonetta Montemagni, and Maria Simi. 2010. A resource and tool for super-sense tagging of Italian texts. In Nicoletta Calzolari, 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).

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL '98)*, pages 86–90, Montreal, Quebec, Canada, August. Association for Computational Linguistics.

D. M. Bikel, R. Schwartz, and R. M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1).

Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 594–602, Sydney, Australia, July. Association for Computational Linguistics.

Massimiliano Ciaramita and Mark Johnson. 2003. Supersense tagging of unknown nouns in WordNet. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 168–175, Sapporo, Japan, July.

James R. Curran. 2005. Supersense tagging of unknown nouns using semantic similarity. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics (ACL'05)*, pages 26–33, Ann Arbor, Michigan, June.

Bonnie J. Dorr, Rebecca J. Passonneau, David Farwell, Rebecca Green, Nizar Habash, Stephen Helmreich, Eduard Hovy, Lori Levin, Keith J. Miller, Teruko Mitamura, Owen Rambow, and Advaith Siddharthan. 2010. Interlingual annotation of parallel text corpora: a new framework for annotation and evaluation. *Natural Language Engineering*, 16(03):197–243.

Sabri Elkateb, William Black, Horacio Rodríguez, Musa Alkhalifa, Piek Vossen, Adam Pease, and Christiane Fellbaum. 2006. Building a WordNet for Arabic. In *Proceedings of The Fifth International Conference on Language Resources and Evaluation (LREC 2006)*, pages 29–34, Genoa, Italy.

Christiane Fellbaum. 1990. English verbs as a semantic net. *International Journal of Lexicography*, 3(4):278–301, December.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. MIT Press, Cambridge, MA.

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 NAACL (HLT-NAACL)*, pages 57–60, New York City, USA, June. Association for Computational Linguistics.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-02)*, Las Palmas, Canary Islands, May.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02)*, pages 1–7, Taipei, Taiwan, August. Association for Computational Linguistics.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proceedings of the Workshop on Human Language Technology (HLT '93)*, HLT '93, pages 303–308, Plainsboro, NJ, USA, March. Association for Computational Linguistics.

George A. Miller. 1990. Nouns in WordNet: a lexical inheritance system. *International Journal of Lexicography*, 3(4):245–264, December.

Behrang Mohit, Nathan Schneider, Rishav Bhowmick, Kemal Oflazer, and Noah A. Smith. 2012. Recall-oriented learning of named entities in Arabic Wikipedia. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2012)*, pages 162–173, Avignon, France, April. Association for Computational Linguistics.

Gerhard Paaß and Frank Reichartz. 2009. Exploiting semantic constraints for estimating supersenses with CRFs. In *Proceedings of the Ninth SIAM International Conference on Data Mining*, pages 485–496, Sparks, Nevada, USA, May. Society for Industrial and Applied Mathematics.

Rebecca J. Passonneau, Ansaf Salleb-Aoussi, Vikas Bhardwaj, and Nancy Ide. 2010. Word sense annotation of polysemous words by multiple annotators. In Nicoletta Calzolari, 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).

Andrew G. Philpot, Michael Fleischman, and Eduard H. Hovy. 2003. Semi-automatic construction of a general purpose ontology. In *Proceedings of the International Lisp Conference*, New York, NY, USA, October.

Davide Picca, Alfio Massimiliano Gliozzo, and Massimiliano Ciaramita. 2008. Supersense Tagger for Italian. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odjik, Stelios Piperidis, and Daniel Tapias, editors, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, pages 2386–2390, Marrakech, Morocco, May. European Language Resources Association (ELRA).

Davide Picca, Alfio Massimiliano Gliozzo, and Simone Campora. 2009. Bridging languages by SuperSense entity tagging. In *Proceedings of the 2009 Named Entities Workshop: Shared Task on Transliteration (NEWS 2009)*, pages 136–142, Suntec, Singapore, August. Association for Computational Linguistics.

Likun Qiu, Yunfang Wu, Yanqiu Shao, and Alexander Gelbukh. 2011. Combining contextual and structural information for supersense tagging of Chinese unknown words. In *Computational Linguistics and Intelligent Text Processing: Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing'11)*, volume 6608 of *Lecture Notes in Computer Science*, pages 15–28. Springer, Berlin.

Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. 2002. Extended named entity hierarchy. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-02)*, Las Palmas, Canary Islands, May.

Annie Zaenen, Jean Carletta, Gregory Garretson, Joan Bresnan, Andrew Koontz-Garboden, Tatiana Nikitina, M. Catherine O'Connor, and Tom Wasow. 2004. Animacy encoding in English: why and how. In Bonnie Webber and Donna K. Byron, editors, *ACL 2004 Workshop on Discourse Annotation*, pages 118–125, Barcelona, Spain, July. Association for Computational Linguistics.

Word Epoch Disambiguation: Finding How Words Change Over Time

Rada Mihalcea
Computer Science and Engineering
University of North Texas
rada@cs.unt.edu

Vivi Nastase
Institute for Computational Linguistics
University of Heidelberg
nastase@cl.uni-heidelberg.de

Abstract

In this paper we introduce the novel task of "word epoch disambiguation," defined as the problem of identifying changes in word usage over time. Through experiments run using word usage examples collected from three major periods of time (1800, 1900, 2000), we show that the task is feasible, and significant differences can be observed between occurrences of words in different periods of time.

1 Introduction

Most current natural language processing works with language as if it were a constant. This however, is not the case. Language is continually changing: we discard or coin new senses for old words; metaphoric and metonymic usages become so engrained that at some point they are considered literal; and we constantly add new words to our vocabulary. The purpose of the current work is to look at language as an evolutionary phenomenon, which we can investigate and analyze and use when working with text collections that span a wide time frame.

Until recently, such task would not have been possible because of the lack of large amounts of non-contemporary data.[1] This has changed thanks to the Google books and Google Ngrams historical projects. They make available in electronic format a large amount of textual data starting from the 17th century, as well as statistics on word usage. We will exploit this data to find differences in word usage across wide periods of time.

[1]While the Brown corpus does include documents from different years, it is far from the scale and time range of Google books.

The phenomena involved in language change are numerous, and for now we focus on word usage in different time epochs. As an example, the word *gay*, currently most frequently used to refer to a sexual orientation, was in the previous century used to express an emotion. The word *run*, in the past used intransitively, has acquired a transitive sense, common in computational circles where we run processes, programs and such.

The purpose of the current research is to quantify changes in word usage, which can be the effect of various factors: changes in meaning (addition/removal of senses), changes in distribution, change in topics that co-occur more frequently with a given word, changes in word spelling, etc. For now we test whether we can identify the epoch to which a word occurrence belongs. We use two sets of words – one with monosemous words, the other with polysemous ones – to try and separate the effect of topic change over time from the effect of sense change.

We use examples from Google books, split into three epochs: 1800+/-25 years, 1900+/-25, 2000+/-25. We select open-class words that occur frequently in all these epochs, and words that occur frequently only in one of them. We then treat each epoch as a "class," and verify whether we can correctly predict this class for test instances from each epoch for the words in our lists. To test whether word usage frequency or sense variation have an impact on this disambiguation task, we use lists of words that have different frequencies in different epochs as well as different polysemies. As mentioned before, we also compare the performance of monosemous – and thus (sensewise) unchanged through time – and polysemous words, to verify whether we can in fact predict sense change as opposed to contextual variation.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 259–263,
Jeju, Republic of Korea, 8-14 July 2012.

2 Related Work

The purpose of this paper is to look at words and how they change in time. Previous work that looks at diachronic language change works at a higher language level, and is not specifically concerned with how words themselves change.

The historical data provided by Google has quickly attracted researchers in various fields, and started the new field of *culturomics* (Michel et al., 2011). The purpose of such research is to analyse changes in human culture, as evidenced by the rise and fall in usage of various terms.

Reali and Griffiths (2010) analyse the similarities between language and genetic evolution, with the transmission of frequency distributions over linguistic forms functioning as the mechanism behind the phenomenon of language change.

Blei and Lafferty (2006) and Blei and Lafferty (2007) track changes in scientific topics through a discrete dynamic topic model (dDTM) – both as types of scientific topics at different time points, and as changing word probability distributions within these topics. The "Photography" topic for example has changed dramatically since the beginning of the 20th century, with words related to digital photography appearing recently, and dominating the most current version of the topic.

Wang and McCallum (2006), Wang et al. (2008) develop time-specific topic models, where topics, as patterns of word use, are tracked across a time changing text collection, and address the task of (fine-grained) time stamp prediction.

Wijaya and Yeniterzi (2011) investigate through topic models the change in context of a specific entity over time, based on the Google Ngram corpus. They determine that changes in this context reflect events occurring in the same period of time.

3 Word Epoch Disambiguation

We formulate the task as a disambiguation problem, where we automatically classify the period of time when a word was used, based on its surrounding context. We use a data-driven formulation, and draw examples from word occurrences over three different epochs. For the purpose of this work, we consider an epoch to be a period of 50 years surrounding the beginning of a new century (1800+/-25 years, 1900+/-25, 2000+/-25). The word usage examples are gathered from books, where the publication year of a book is judged to be representative for the time when that word was used. We select words with different characteristics to allow us to investigate whether there is an effect caused by sense change, or the disambiguation performance comes from the change of topics and vocabulary over time.

4 Experimental Setting

Target Words. The choice of target words for our experiments is driven by the phenomena we aim to analyze. Because we want to investigate the behavior of words in different epochs, and verify whether the difference in word behavior comes from changes in sense or changes in wording in the context, we choose a mixture of polysemous words and monosemous words (according to WordNet and manually checked against Webster's dictionary editions from 1828, 1913 and the current Merriam-Webster edition), and also words that are frequent in all epochs, as well as words that are frequent in only one epoch.

According to these criteria, for each open class (nouns, verbs, adjectives, adverbs) we select 50 words, 25 of which have multiple senses, 25 with one sense only. Each of these two sets has a 10-5-5-5 distribution: 10 words that are frequent in all three epochs, and 5 per each epoch such that these words are only frequent in one epoch. To avoid part-of-speech ambiguity we also choose words that are unambiguous from this point of view. This selection process was done based on Google 1gram historical data, used for computing the probability distribution of open-class words for each epoch. [2]

The set of target words consists thus of 200 open class words, uniformly distributed over the 4 parts of speech, uniformly distributed over multiple-sense/unique sense words, and with the frequency based sample as described above. From this initial set of words, we could not identify enough examples in the three epochs considered for 35,[3] which left us with a final set of 165 words.

Data. For each target word in our dataset, we collect the top 100 snippets returned by a search on Google Books for each of the three epochs we consider.

[2] For each open class word we create ranked lists of words, where the ranking score is an adjusted *tfidf* score – the epochs correspond to documents. To choose words frequent only in one epoch, we choose the top words in the list, for words frequent in all epochs we choose the bottom words in this list.

[3] A minimum of 30 total examples was required for a word to be considered in the dataset.

All the extracted snippets are then processed: the text is tokenized and part-of-speech tagged using the Stanford tagger (Toutanova et al., 2003), and contexts that do not include the target word with the specified part-of-speech are removed. The position of the target word is also identified and recorded as an offset along with the example.

For illustration, we show below an example drawn from each epoch for two different words, *dinner*:

> 1800: On reaching Mr. Crane's house, dinner was set before us ; but as is usual here in many places on the Sabbath, it was both **dinner** and tea combined into a single meal.
> 1900: The average **dinner** of today consists of relishes; of soup, either a consomme (clear soup) or a thick soup.
> 2000: Preparing **dinner** in a slow cooker is easy and convenient because the meal you're making requires little to no attention while it cooks.

and *surgeon*:

> 1800: The apothecaries must instantly dispense what medicines the **surgeons** require for the use of the regiments.
> 1900: The **surgeon** operates, collects a fee, and sends to the physician one-third or one-half of the fee, this last transaction being unknown to the patient.
> 2000: From a New York plastic surgeon comes all anyone ever wanted to know–and never imagined–about what goes on behind the scenes at the office of one of the world's most prestigious plastic **surgeons**.

Disambiguation Algorithm. The classification algorithm we use is inspired by previous work on data-driven word sense disambiguation. Specifically, we use a system that integrates both local and topical features. The *local features* include: the current word and its part-of-speech; a local context of three words to the left and right of the ambiguous word; the parts-of-speech of the surrounding words; the first noun before and after the target word; the first verb before and after the target word. The *topical features* are determined from the global context and are implemented through class-specific keywords, which are determined as a list of at most five words occurring at least three times in the contexts defining a certain word class (or epoch). This feature set is similar to the one used by (Ng and Lee, 1996). The features are then integrated in a Naive Bayes classifier (Lee and Ng, 2002).

POS	No. words	Avg. no. examples	Baseline	WED
Noun	46	190	42.54%	66.17%
Verb	49	198	42.25%	59.71%
Adjective	26	136	48.60%	60.13%
Adverb	44	213	40.86%	59.61%
AVERAGE	165	190	42.96%	61.55%

Table 1: Overall results for different parts-of-speech.

Evaluation. To evaluate word epoch disambiguation, we calculate the average accuracy obtained through ten-fold cross-validations applied on the data collected for each word. To place results in perspective, we also calculate a simple baseline, which assigns the most frequent class by default.

5 Results and Discussion

Table 1 summarizes the results obtained for the 165 words. Overall, the task appears to be feasible, as absolute improvements of 18.5% are observed. While improvements are obtained for all parts-of-speech, the nouns lead to the highest disambiguation results, with the largest improvement over the baseline, which interestingly aligns with previous observations from work on word sense disambiguation (Mihalcea and Edmonds, 2004; Agirre et al., 2007).

Among the words considered, there are words that experience very large improvements over the baseline, such as "computer" (with an absolute increase over the baseline of 42%) or "install" (41%), which are words that are predominantly used in one of the epochs considered (2000), and are also known to have changed meaning over time. There are also words that experience very small improvements, such as "again" (3%) or "captivate" (7%), which are words that are frequently used in all three epochs. There are even a few words (seven) for which the disambiguation accuracy is below the baseline, such as "oblige" (-1%) or "cruel" (-15%).

To understand to what extent the change in frequency over time has an impact on word epoch disambiguation, in Table 2 we report results for words that have high frequency in all three epochs considered, or in only one epoch at a time. As expected, the words that are used more often in an epoch are also easier to disambiguate.[4] For instance, the

[4]The difference in results does not come from difference in

verb "reassert" has higher frequency in 2000, and it has a disambiguation accuracy of 67.25% compared to a baseline of 34.15%. Instead, the verb "conceal," which appears with high frequency in all three epochs, has a disambiguation accuracy of 44.70%, which is a relatively small improvement over the baseline of 38.04%.

POS	No. words	Avg. no. examples	Baseline	WED
High frequency in all epochs				
Noun	18	180	42.31%	65.77%
Verb	19	203	43.45%	56.43%
Adjective	7	108	46.27%	57.75%
Adverb	17	214	40.32%	56.41%
AVERAGE	61	188	42.56%	59.33%
High frequency in one epoch				
Noun	28	196	42.68%	66.42%
Verb	30	194	41.50%	61.80%
Adjective	19	146	49.47%	61.02%
Adverb	27	213	41.20%	61.63%
AVERAGE	104	191	43.20%	62.86%

Table 2: Results for words that have high frequency in all epochs, or in one epoch at a time

The second analysis that we perform is concerned with the accuracy observed for polysemous words as compared to monosemous words. Comparative results are reported in Table 3. Monosemous words do not have sense changes over time, so being able to classify them in different epochs relies exclusively on variations in their context over time. Polysemous words's context change because of both changes in topics/vocabulary over time, and changes in word senses. The fact that we see a difference in accuracy between disambiguation results for monosemous and polysemous words is an indication that word sense change is reflected and can be captured in the context.

To better visualize the improvements obtained with word epoch disambiguation with respect to the baseline, Figure 1 plots the results.

6 Conclusions

In this paper, we introduced the novel task of word epoch disambiguation, which aims to quantify the changes in word usage over time. Using examples collected from three major periods of time, for 165 words, we showed that the word epoch disambiguation algorithm can lead to an overall absolute improvement of 18.5%, as compared to a baseline that picks the most frequent class by default. These results indicate that there are significant differences between occurrences of words in different periods of time. Moreover, additional analyses suggest that changes in usage frequency and word senses contribute to these differences. In future work, we plan to do an in-depth analysis of the features that best characterize the changes in word usage over time, and develop representations that allow us to track sense changes.

size in the data, as the number of examples extracted for words of high or low frequency is approximately the same.

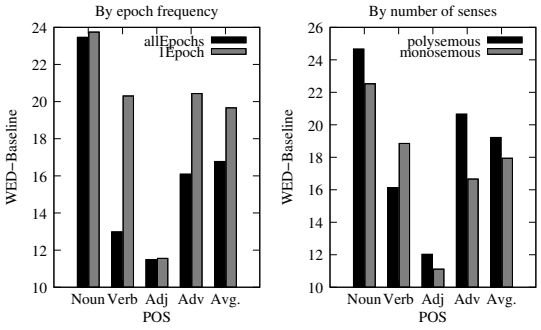


Figure 1: Word epoch disambiguation compared to the baseline, for words that are frequent/not frequent (in a given epoch), and monosemous/polysemous.

POS	No. words	Avg. no. examples	Baseline	WED
Polysemous words				
Noun	24	191	41.89%	66.55%
Verb	25	214	42.71%	58.84%
Adjective	12	136	45.40%	57.42%
Adverb	23	214	39.38%	60.03%
AVERAGE	84	196	41.94%	61.16%
Monosemous words				
Noun	22	188	43.25%	65.77%
Verb	24	181	41.78%	60.63%
Adjective	14	136	51.36%	62.47%
Adverb	21	213	42.49%	59.15%
AVERAGE	81	183	44.02%	61.96%

Table 3: Results for words that are polysemous or monosemous.

Acknowledgments

This material is based in part upon work supported by the National Science Foundation CAREER award #0747340. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

E. Agirre, L. Marquez, and R. Wicentowski, editors. 2007. *Proceedings of the 4th International Workshop on Semantic Evaluations*, Prague, Czech Republic.

D. Blei and J. Lafferty. 2006. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*.

D. Blei and J. Lafferty. 2007. A correlated topic model of Science. *The Annals of Applied Science*, 1(1):17–35.

Y.K. Lee and H.T. Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, Philadelphia, June.

J.-B. Michel, Y.K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, S. Pinker, M. A. Nowak, and E. L. Aiden. 2011. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, January.

R. Mihalcea and P. Edmonds, editors. 2004. *Proceedings of SENSEVAL-3, Association for Computational Linguistics Workshop*, Barcelona, Spain.

H.T. Ng and H.B. Lee. 1996. Integrating multiple knowledge sources to disambiguate word sense: An examplar-based approach. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996)*, Santa Cruz.

F. Reali and T. Griffiths. 2010. Words as alleles: connecting language evolution with bayesian learners to models of genetic drift. *Proceedings of the Royal Society*, 277(1680):429–436.

K. Toutanova, D. Klein, C. Manning, and Y. Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of Human Language Technology Conference (HLT-NAACL 2003)*, Edmonton, Canada, May.

X. Wang and A. McCallum. 2006. Topics over time: A non-Markov continuous-time model of topical trends. In *Conference on Knowledge Discovery and Data Mining (KDD)*.

C. Wang, D. Blei, and D. Heckerman. 2008. Continuous time dynamic topic models. In *International Conference on Machine Learning (ICML)*.

D. Wijaya and R. Yeniterzi. 2011. Understanding semantic change of words over centuries. In *Proc. of the Workshop on Detecting and Exploiting Cultural Diversity on the Social Web (DETECT) 2011*.

Authorship Attribution with Author-aware Topic Models

Yanir Seroussi **Fabian Bohnert** **Ingrid Zukerman**

Faculty of Information Technology, Monash University
Clayton, Victoria 3800, Australia
firstname.lastname@monash.edu

Abstract

Authorship attribution deals with identifying the authors of anonymous texts. Building on our earlier finding that the Latent Dirichlet Allocation (LDA) topic model can be used to improve authorship attribution accuracy, we show that employing a previously-suggested Author-Topic (AT) model outperforms LDA when applied to scenarios with many authors. In addition, we define a model that combines LDA and AT by representing authors and documents over two disjoint topic sets, and show that our model outperforms LDA, AT and support vector machines on datasets with many authors.

1 Introduction

Authorship attribution (AA) has attracted much attention due to its many applications in, e.g., computer forensics, criminal law, military intelligence, and humanities research (Stamatatos, 2009). The traditional problem, which is the focus of our work, is to attribute *test texts* of unknown authorship to one of a set of known authors, whose *training texts* are supplied in advance (i.e., a supervised classification problem). While most of the early work on AA focused on formal texts with only a few possible authors, researchers have recently turned their attention to informal texts and tens to thousands of authors (Koppel et al., 2011). In parallel, topic models have gained popularity as a means of analysing such large text corpora (Blei, 2012). In (Seroussi et al., 2011), we showed that methods based on *Latent Dirichlet Allocation* (LDA) – a popular topic model by Blei et al. (2003) – yield good AA performance. However, LDA does not model authors explicitly, and we are not aware of any previous studies that apply *author-aware* topic models to traditional AA. This paper aims to address this gap.

In addition to being the first (to the best of our knowledge) to apply Rosen-Zvi et al.'s (2004) *Author-Topic Model* (AT) to traditional AA, the main contribution of this paper is our *Disjoint Author-Document Topic Model* (DADT), which addresses AT's limitations in the context of AA. We show that DADT outperforms AT, LDA, and linear support vector machines on AA with many authors.

2 Disjoint Author-Document Topic Model

Background. Our definition of DADT is motivated by the observation that when authors write texts on the same issue, specific words must be used (e.g., texts about LDA are likely to contain the words "topic" and "prior"), while other words vary in frequency according to author style. Also, texts by the same author share similar style markers, independently of content (Koppel et al., 2009). DADT aims to separate *document words* from *author words* by generating them from two disjoint topic sets of $T^{(D)}$ *document topics* and $T^{(A)}$ *author topics*.

Lacoste-Julien et al. (2008) and Ramage et al. (2009) (among others) also used disjoint topic sets to represent document labels, and Chemudugunta et al. (2006) separated corpus-level topics from document-specific words. However, we are unaware of any applications of these ideas to AA. The closest work we know of is by Mimno and McCallum (2008), whose DMR model outperformed AT in AA

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 264–269,
Jeju, Republic of Korea, 8-14 July 2012.

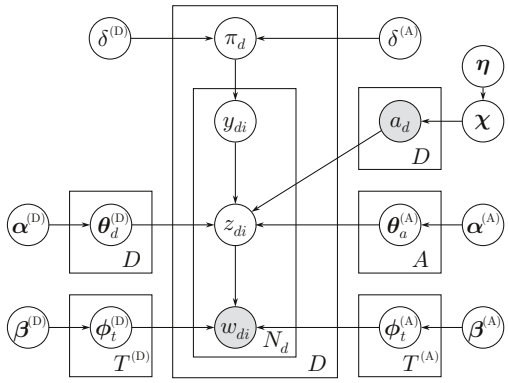


Figure 1: The Disjoint Author-Document Topic Model

of *multi-authored* texts (DMR does not use disjoint topic sets). We use AT rather than DMR, since we found that AT outperforms DMR in AA of *single-authored* texts, which are the focus of this paper.

The Model. Figure 1 shows DADT's graphical representation, with document-related parameters on the left (the LDA component), and author-related parameters on the right (the AT component). We define the model for single-authored texts, but it can be easily extended to multi-authored texts.

The generative process for DADT is described below. We use $\mathcal{D}$ and $\mathcal{C}$ to denote the Dirichlet and categorical distributions respectively, and A, D and V to denote the number of authors, documents, and unique vocabulary words respectively. In addition, we mark each step as coming from either **L**DA or **A**T, or as new in **D**ADT.

Global level:

- **L.** For each document topic t, draw a word distribution $\boldsymbol{\phi}_t^{(D)} \sim \mathcal{D}\left(\boldsymbol{\beta}^{(D)}\right)$, where $\boldsymbol{\beta}^{(D)}$ is a length-V vector.
- **A.** For each author topic t, draw a word distribution $\boldsymbol{\phi}_t^{(A)} \sim \mathcal{D}\left(\boldsymbol{\beta}^{(A)}\right)$, where $\boldsymbol{\beta}^{(A)}$ is a length-V vector.
- **A.** For each author a, draw the author topic distribution $\boldsymbol{\theta}_a^{(A)} \sim \mathcal{D}\left(\boldsymbol{\alpha}^{(A)}\right)$, where $\boldsymbol{\alpha}^{(A)}$ is a length-$T^{(A)}$ vector.
- **D.** Draw a distribution over authors $\boldsymbol{\chi} \sim \mathcal{D}\left(\boldsymbol{\eta}\right)$, where $\boldsymbol{\eta}$ is a length-A vector.

Document level: For each document d:

- **L.** Draw d's topic distribution $\boldsymbol{\theta}_d^{(D)} \sim \mathcal{D}\left(\boldsymbol{\alpha}^{(D)}\right)$, where $\boldsymbol{\alpha}^{(D)}$ is a length-$T^{(D)}$ vector.
- **D.** Draw d's author $a_d \sim \mathcal{C}\left(\boldsymbol{\chi}\right)$.
- **D.** Draw d's topic ratio $\pi_d \sim \text{Beta}\left(\delta^{(A)}, \delta^{(D)}\right)$, where $\delta^{(A)}$ and $\delta^{(D)}$ are scalars.

Word level: For each word index i in document d:

- **D.** Draw di's topic indicator $y_{di} \sim \text{Bernoulli}(\pi_d)$.
- **L.** If $y_{di} = 0$, draw a *document* topic $z_{di} \sim \mathcal{C}\left(\boldsymbol{\theta}_d^{(D)}\right)$ and word $w_{di} \sim \mathcal{C}\left(\boldsymbol{\phi}_{z_{di}}^{(D)}\right)$.
- **A.** If $y_{di} = 1$, draw an *author* topic $z_{di} \sim \mathcal{C}\left(\boldsymbol{\theta}_{a_d}^{(A)}\right)$ and word $w_{di} \sim \mathcal{C}\left(\boldsymbol{\phi}_{z_{di}}^{(A)}\right)$.

DADT versus AT. DADT might seem similar to AT with "fictitious" authors, as described by Rosen-Zvi et al. (2010) (i.e., AT trained with an additional unique "fictitious" author for each document, allowing it to adapt to individual documents and not only to authors). However, there are several key differences between DADT and AT.

First, in DADT *author topics are disjoint from document topics*, with different priors for each topic set. Thus, the number of author topics can be different from the number of document topics, enabling us to vary the number of author topics according to the number of authors in the corpus.

Second, DADT *places different priors on the word distributions* for author topics and document topics ($\boldsymbol{\beta}^{(A)}$ and $\boldsymbol{\beta}^{(D)}$ respectively). Stopwords are known to be strong indicators of authorship (Koppel et al., 2009), and DADT allows us to use this knowledge by assigning higher weights to the elements of $\boldsymbol{\beta}^{(A)}$ that correspond to stopwords than to such elements in $\boldsymbol{\beta}^{(D)}$.

Third, DADT *learns the ratio between document words and author words* on a per-document basis, and makes it possible to specify a prior belief of what this ratio should be. We found that specifying a prior belief that about 80% of each document is composed of author words yielded better results than using AT's approach, which evenly splits each document into author and document words.

Fourth, DADT *defines the process that generates authors*. This allows us to consider the number of texts by each author when performing AA. This also enables the potential use of DADT in a semi-supervised setting by training on unlabelled texts, which we plan to explore in the future.

3 Authorship Attribution Methods

We experimented with the following AA methods, using token frequency features, which are good predictors of authorship (Koppel et al., 2009).

Baseline: Support Vector Machines (SVMs). Koppel et al. (2009) showed that SVMs yield good AA performance. We use linear SVMs in a one-versus-all setup, as implemented in LIBLINEAR (Fan et al., 2008), reporting results obtained with the best cost parameter values.

Baseline: LDA + Hellinger (LDA-H). This approach uses the Hellinger distances of topic distributions to assign test texts to the closest author. In (Seroussi et al., 2011), we experimented with two variants: (1) each author's texts are concatenated before building the LDA model; and (2) no concatenation is performed. We found that the latter approach performs poorly in cases with many candidate authors. Hence, we use only the former approach in this paper. Note that when dealing with single-authored texts, concatenating each author's texts yields an LDA model that is equivalent to AT.

AT. Given an inferred AT model (Rosen-Zvi et al., 2004), we calculate the probability of the test text words for each author a, assuming it was written by a, and return the most probable author. We do not know of any other studies that used AT in this manner for single-authored AA. We expect this method to outperform LDA-H as it employs AT directly, rather than relying on an external distance measure.

AT-FA. Same as AT, but built with an additional unique "fictitious" author for each document.

DADT. Given our DADT model, we assume that the test text was written by a "new" author, and infer this author's topic distribution, the author/document topic ratio, and the document topic distribution. We then calculate the probability of each author given the model's parameters, the test text words, and the inferred author/document topic ratio and document topic distribution. The most probable author is returned. We use this method to avoid inferring the document-dependent parameters separately for each author, which is infeasible when many authors exist. A version that marginalises over these parameters will be explored in future work.

4 Evaluation

We compare the performance of the methods on two publicly-available datasets: (1) *PAN'11*: emails with 72 authors (Argamon and Juola, 2011); and (2) *Blog:* blogs with 19,320 authors (Schler et al., 2006). These datasets represent realistic scenarios of AA of user-generated texts with many candidate authors. For example, Chaski (2005) notes a case where an employee who was terminated for sending a racist email claimed that any person with access to his computer could have sent the email.

Experimental Setup. Experiments on the PAN'11 dataset followed the setup of the PAN'11 competition (Argamon and Juola, 2011): We trained all the methods on the given training subset, tuned the parameters according to the results on the given validation subset, and ran the tuned methods on the given testing subset. In the Blog experiments, we used ten-fold cross validation as in (Seroussi et al., 2011).

We used collapsed Gibbs sampling to train all the topic models (Griffiths and Steyvers, 2004), running 4 chains with a burn-in of 1,000 iterations. In the PAN'11 experiments, we retained 8 samples per chain with spacing of 100 iterations. In the Blog experiments, we retained 1 sample per chain due to runtime constraints. Since we cannot average topic distribution estimates obtained from training samples due to topic exchangeability (Steyvers and Griffiths, 2007), we averaged the distances and probabilities calculated from the retained samples. For test text sampling, we used a burn-in of 100 iterations and averaged the parameter estimates over the next 100 iterations in a similar manner to Rosen-Zvi et al. (2010). We found that these settings yield stable results across different random seed values.

We found that the number of topics has a larger impact on accuracy than other configurable parameters. Hence, we used symmetric topic priors, setting all the elements of $\boldsymbol{\alpha}^{(D)}$ and $\boldsymbol{\alpha}^{(A)}$ to $\min\{0.1, 5/T^{(D)}\}$ and $\min\{0.1, 5/T^{(A)}\}$ respectively.[1] For all models, we set $\beta_w = 0.01$ for each word w as the base measure for the prior of words in topics. Since DADT allows us to encode our prior knowledge that stopword use is indicative of authorship, we set $\beta_w^{(D)} = 0.01 - \epsilon$ and $\beta_w^{(A)} = 0.01 + \epsilon$ for all w, where w is a stopword.[2] We set $\epsilon = 0.009$, which improved accuracy by up to one percentage point over using $\epsilon = 0$. Finally, we set $\delta^{(A)} = 4.889$ and $\delta^{(D)} = 1.222$ for DADT. This encodes our prior

[1] We tested Wallach et al.'s (2009) method of obtaining asymmetric priors, but found that it did not improve accuracy.

[2] We used the stopword list from `www.lextek.com/manuals/onix/stopwords2.html`.

Method	PAN'11 Validation	PAN'11 Testing	Blog Prolific	Blog Full
SVM	48.61%	53.31%	33.31%	24.13%
LDA-H	34.95%	42.62%	21.61%	7.94%
AT	46.68%	53.08%	37.56%	23.03%
AT-FA	20.68%	24.23%	—	—
DADT	**54.24%**	**59.08%**	**42.51%**	**27.63%**

Table 1: Experiment results

belief that 0.8 ± 0.15 of each document is composed of author words. We found that this yields better results than an uninformed uniform prior of $\delta^{(A)} = \delta^{(D)} = 1$ (Seroussi et al., 2012). In addition, we set $\eta_a = 1$ for each author a, yielding smoothed estimates for the corpus distribution of authors $\boldsymbol{\chi}$.

To fairly compare the topic-based methods, we used the same overall number of topics for all the topic models. We present only the results obtained with the best topic settings: 100 for PAN'11 and 400 for Blog, with DADT's author/document topic splits being 90/10 for PAN'11, and 390/10 for Blog. These splits allow DADT to de-noise the author representations by allocating document words to a relatively small number of document topics. It is worth noting that AT can be seen as an extreme version of DADT, where all the topics are author topics. A future extension is to learn the topic balance automatically, e.g., in a similar manner to Teh et al.'s (2006) method of inferring the number of topics in LDA.

Results. Table 1 shows the results of our experiments in terms of classification accuracy (i.e., the percentage of test texts correctly attributed to their author). The PAN'11 results are shown for the validation and testing subsets, and the Blog results are shown for a subset containing the 1,000 most prolific authors and for the full dataset of 19,320 authors.

Our DADT model yielded the best results in all cases (the differences between DADT and the other methods are statistically significant according to a paired two-tailed t-test with $p < 0.05$). We attribute DADT's superior performance to the de-noising effect of the disjoint topic sets, which appear to yield author representations of higher predictive quality than those of the other models.

As expected, AT significantly outperformed LDA-H. On the other hand, AT-FA performed much worse than all the other methods on PAN'11, probably because of the inherent noisiness in using the same topics to model both authors and documents. Hence, we did not run AT-FA on the Blog dataset.

DADT's PAN'11 testing result is close to the third-best accuracy from the PAN'11 competition (Argamon and Juola, 2011). However, to the best of our knowledge, DADT obtained the best accuracy for a fully-supervised method that uses only unigram features. Specifically, Kourtis and Stamatatos (2011), who obtained the highest accuracy (65.8%), assumed that all the test texts are given to the classifier at the same time, and used this additional information with a semi-supervised method; while Kern et al. (2011) and Tanguy et al. (2011), who obtained the second-best (64.2%) and third-best (59.4%) accuracies respectively, used various feature types (e.g., features obtained from parse trees). Further, preprocessing differences make it hard to compare the methods on a level playing field. Nonetheless, we note that extending DADT to enable semi-supervised classification and additional feature types are promising future work directions.

While all the methods yielded relatively low accuracies on Blog due to its size, topic-based methods were more strongly affected than SVM by the transition from the 1,000 author subset to the full dataset. This is probably because topic-based methods use a single model, making them more sensitive to corpus size than SVM's one-versus-all setup that uses one model per author. Notably, an oracle that chooses the correct answer between SVM and DADT when they disagree yields an accuracy of 37.15% on the full dataset, suggesting it is worthwhile to explore ensembles that combine the outputs of SVM and DADT (we tried using DADT topics as additional SVM features, but this did not outperform DADT).

5 Conclusion

This paper demonstrated the utility of using author-aware topic models for AA: AT outperformed LDA, and our DADT model outperformed LDA, AT and SVMs in cases with noisy texts and many authors. We hope that these results will inspire further research into the application of topic models to AA.

Acknowledgements

This research was supported in part by Australian Research Council grant LP0883416. We thank Mark Carman for fruitful discussions on topic modelling.

References

Shlomo Argamon and Patrick Juola. 2011. Overview of the international authorship identification competition at PAN-2011. In *CLEF 2011: Proceedings of the 2011 Conference on Multilingual and Multimodal Information Access Evaluation (Lab and Workshop Notebook Papers)*, Amsterdam, The Netherlands.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.

David M. Blei. 2012. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.

Carole E. Chaski. 2005. Who's at the keyboard? Authorship attribution in digital evidence investigations. *International Journal of Digital Evidence*, 4(1).

Chaitanya Chemudugunta, Padhraic Smyth, and Mark Steyvers. 2006. Modeling general and specific aspects of documents with a probabilistic topic model. In *NIPS 2006: Proceedings of the 20th Annual Conference on Neural Information Processing Systems*, pages 241–248, Vancouver, BC, Canada.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9(Aug):1871–1874.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(Suppl. 1):5228–5235.

Roman Kern, Christin Seifert, Mario Zechner, and Michael Granitzer. 2011. Vote/veto meta-classifier for authorship identification. In *CLEF 2011: Proceedings of the 2011 Conference on Multilingual and Multimodal Information Access Evaluation (Lab and Workshop Notebook Papers)*, Amsterdam, The Netherlands.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2009. Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, 60(1):9–26.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2011. Authorship attribution in the wild. *Language Resources and Evaluation*, 45(1):83–94.

Ioannis Kourtis and Efstathios Stamatatos. 2011. Author identification using semi-supervised learning. In *CLEF 2011: Proceedings of the 2011 Conference on Multilingual and Multimodal Information Access Evaluation (Lab and Workshop Notebook Papers)*, Amsterdam, The Netherlands.

Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. 2008. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *NIPS 2008: Proceedings of the 22nd Annual Conference on Neural Information Processing Systems*, pages 897–904, Vancouver, BC, Canada.

David Mimno and Andrew McCallum. 2008. Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. In *UAI 2008: Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, pages 411–418, Helsinki, Finland.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *EMNLP 2009: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256, Singapore.

Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. In *UAI 2004: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 487–494, Banff, AB, Canada.

Michal Rosen-Zvi, Chaitanya Chemudugunta, Thomas Griffiths, Padhraic Smyth, and Mark Steyvers. 2010. Learning author-topic models from text corpora. *ACM Transactions on Information Systems*, 28(1):1–38.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W. Pennebaker. 2006. Effects of age and gender on blogging. In *Proceedings of AAAI Spring Symposium on Computational Approaches for Analyzing Weblogs*, pages 199–205, Stanford, CA, USA.

Yanir Seroussi, Ingrid Zukerman, and Fabian Bohnert. 2011. Authorship attribution with latent Dirichlet allocation. In *CoNLL 2011: Proceedings of the 15th International Conference on Computational Natural Language Learning*, pages 181–189, Portland, OR, USA.

Yanir Seroussi, Fabian Bohnert, and Ingrid Zukerman. 2012. Authorship attribution with author-aware topic models. Technical Report 2012/268, Faculty of Information Technology, Monash University, Clayton, VIC, Australia.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556.

Mark Steyvers and Tom Griffiths. 2007. Probabilistic topic models. In Thomas K. Landauer, Danielle S. McNamara, Simon Dennis, and Walter Kintsch, editors, *Handbook of Latent Semantic Analysis*, pages 427–448. Lawrence Erlbaum Associates.

Ludovic Tanguy, Assaf Urieli, Basilio Calderone, Nabil Hathout, and Franck Sajous. 2011. A multitude of linguistically-rich features for authorship attribution. In *CLEF 2011: Proceedings of the 2011 Conference on Multilingual and Multimodal Information Access Evaluation (Lab and Workshop Notebook Papers)*, Amsterdam, The Netherlands.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. 2006. Hierarchical Dirichlet pro-

cesses. *Journal of the American Statistical Association*, 101(476):1566–1581.

Hanna M. Wallach, David Mimno, and Andrew McCallum. 2009. Rethinking LDA: Why priors matter. In *NIPS 2009: Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, pages 1973–1981, Vancouver, BC, Canada.

Information-theoretic Multi-view Domain Adaptation

Pei Yang[1,3], **Wei Gao**[2], **Qi Tan**[1], **Kam-Fai Wong**[3]
[1]South China University of Technology, Guangzhou, China
{yangpei,tanqi}@scut.edu.cn
[2]Qatar Computing Research Institute, Qatar Foundation, Doha, Qatar
wgao@qf.org.qa
[3]The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
kfwong@se.cuhk.edu.hk

Abstract

We use multiple views for cross-domain document classification. The main idea is to strengthen the views' consistency for target data with source training data by identifying the correlations of domain-specific features from different domains. We present an Information-theoretic Multi-view Adaptation Model (IMAM) based on a multi-way clustering scheme, where word and link clusters can draw together seemingly unrelated domain-specific features from both sides and iteratively boost the consistency between document clusterings based on word and link views. Experiments show that IMAM significantly outperforms state-of-the-art baselines.

1 Introduction

Domain adaptation has been shown useful to many natural language processing applications including document classification (Sarinnapakorn and Kubat, 2007), sentiment classification (Blitzer et al., 2007), part-of-speech tagging (Jiang and Zhai, 2007) and entity mention detection (Daumé III and Marcu, 2006).

Documents can be represented by multiple independent sets of features such as words and link structures of the documents. Multi-view learning aims to improve classifiers by leveraging the redundancy and consistency among these multiple views (Blum and Mitchell, 1998; Rüping and Scheffer, 2005; Abney, 2002). Existing methods were designed for data from single domain, assuming that either view alone is sufficient to predict the target class accurately. However, this view-consistency assumption is largely violated in the setting of domain adaptation where training and test data are drawn from different distributions.

Little research was done for multi-view domain adaptation. In this work, we present an Information-theoretical Multi-view Adaptation Model (IMAM) based on co-clustering framework (Dhillon et al., 2003) that combines the two learning paradigms to transfer class information across domains in multiple transformed feature spaces. IMAM exploits a multi-way-clustering-based classification scheme to simultaneously cluster documents, words and links into their respective clusters. In particular, the word and link clusterings can automatically associate the correlated features from different domains. Such correlations bridge the domain gap and enhance the consistency of views for clustering (i.e., classifying) the target data. Results show that IMAM significantly outperforms the state-of-the-art baselines.

2 Related Work

The work closely related to ours was done by Dai et al. (2007), where they proposed co-clustering-based classification (CoCC) for adaptation learning. CoCC was extended from information-theoretic co-clustering (Dhillon et al., 2003), where in-domain constraints were added to word clusters to provide the class structure and partial categorization knowledge. However, CoCC is a single-view algorithm.

Although multi-view learning (Blum and Mitchell, 1998; Dasgupta et al., 2001; Abney, 2002; Sridharan and Kakade, 2008) is common within a single domain, it is not well studied under cross-domain settings. Chen et al. (2011) proposed

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 270–274,
Jeju, Republic of Korea, 8-14 July 2012.

CODA for adaptation based on co-training (Blum and Mitchell, 1998), which is however a pseudo multi-view algorithm where original data has only one view. Therefore, it is not suitable for the true multi-view case as ours. Zhang et al. (2011) proposed an instance-level multi-view transfer algorithm that integrates classification loss and view consistency terms based on large margin framework. However, instance-based approach is generally poor since new target features lack support from source data (Blitzer et al., 2011). We focus on feature-level multi-view adaptation.

3 Our Model

Intuitively, source-specific and target-specific features can be drawn together by mining their co-occurrence with domain-independent (common) features, which helps bridge the distribution gap. Meanwhile, the view consistency on target data can be strengthened if target-specific features are appropriately bundled with source-specific features. Our model leverages the complementary cooperation between different views to yield better adaptation performance.

3.1 Representation

Let D_S be the source training documents and D_T be the unlabeled target documents. Let C be the set of class labels. Each source document $d_s \in D_S$ is labeled with a unique class label $c \in C$. Our goal is to assign each target document $d_t \in D_T$ to an appropriate class as accurately as possible.

Let W be the vocabulary of the entire document collection $D = D_S \cup D_T$. Let L be the set of all links (hyperlinks or citations) among documents. Each $d \in D$ can be represented by two views, i.e., a bag-of-words set $\{w\}$ and a bag-of-links set $\{l\}$.

Our model explores multi-way clustering that simultaneously clusters documents, words and links. Let $\hat{D}$, $\hat{W}$ and $\hat{L}$ be the respective clustering of documents, words and links. The clustering functions are defined as $\mathcal{C}_D(d) = \hat{d}$ for document, $\mathcal{C}_W(w) = \hat{w}$ for word and $\mathcal{C}_L(l) = \hat{l}$ for link, where $\hat{d}$, $\hat{w}$ and $\hat{l}$ represent the corresponding clusters.

3.2 Objectives

We extend the information-theoretic co-clustering framework (Dhillon et al., 2003) to incorporate the loss from multiple views. Let $\mathcal{I}(X, Y)$ be mutual information (MI) of variables X and Y, our objective is to minimize the MI loss of two different views:

$$\Theta = \alpha \cdot \Theta_W + (1 - \alpha) \cdot \Theta_L \quad (1)$$

where

$$\Theta_W = \mathcal{I}(D_T, W) - \mathcal{I}(\hat{D}_T, \hat{W}) + \lambda \cdot \Big[\mathcal{I}(C, W) - \mathcal{I}(C, \hat{W})\Big]$$

$$\Theta_L = \mathcal{I}(D_T, L) - \mathcal{I}(\hat{D}_T, \hat{L}) + \lambda \cdot \Big[\mathcal{I}(C, L) - \mathcal{I}(C, \hat{L})\Big]$$

Θ_W and Θ_L are the loss terms based on word view and link view, respectively, traded off by α. λ balances the effect of word or link clusters from co-clustering. When $\alpha = 1$, the function relies on text only that reduces to CoCC (Dai et al., 2007).

For any $x \in \hat{x}$, we define conditional distribution $q(x|\hat{y}) = p(x|\hat{x})p(\hat{x}|\hat{y})$ under co-clustering $(\hat{X}, \hat{Y})$ based on Dhillon et al. (2003). Therefore, for any $w \in \hat{w}$, $l \in \hat{l}$, $d \in \hat{d}$ and $c \in C$, we can calculate a set of conditional distributions: $q(w|\hat{d})$, $q(d|\hat{w})$, $q(l|\hat{d})$, $q(d|\hat{l})$, $q(c|\hat{w})$, $q(c|\hat{l})$.

Eq. 1 is hard to optimize due to its combinatorial nature. We transform it to the equivalent form based on Kullback-Leibler (KL) divergence between two conditional distributions $p(x|y)$ and $q(x|\hat{y})$, where $\mathcal{D}(p(x|y)||q(x|\hat{y})) = \sum_x p(x|y) log \frac{p(x|y)}{q(x|\hat{y})}$.

Lemma 1 (Objective functions) *Equation 1 can be turned into the form of alternate minimization: (i) For document clustering, we minimize*

$$\Theta = \sum_d p(d)\phi_D(d, \hat{d}) + \phi_C(\hat{W}, \hat{L}),$$

where $\phi_C(\hat{W}, \hat{L})$ is a constant[1] *and*

$$\phi_D(d, \hat{d}) = \alpha \cdot \mathcal{D}(p(w|d)||q(w|\hat{d})) + (1 - \alpha) \cdot \mathcal{D}(p(l|d)||q(l|\hat{d})).$$

(ii) For word and link clustering, we minimize

$$\Theta = \alpha \sum_w p(w)\phi_W(w, \hat{w}) + (1-\alpha) \sum_l p(l)\phi_L(l, \hat{l}),$$

where for any feature v (e.g., w or l) in feature set V (e.g., W or L), we have

$$\phi_V(v, \hat{v}) = \mathcal{D}(p(d|v)||q(d|\hat{v})) + \lambda \cdot \mathcal{D}(p(c|v)||q(c|\hat{v})).$$

[1] We can obtain that $\phi_C(\hat{W}, \hat{L}) = \lambda \Big[\alpha(\mathcal{I}(C, W) - \mathcal{I}(C, \hat{W})) + (1 - \alpha)(\mathcal{I}(C, L) - \mathcal{I}(C, \hat{L}))\Big]$, which is constant since word/link clusters keep fixed during the document clustering step.

Lemma 1[2] allows us to *alternately* reorder either documents or both words and links by fixing the other in such a way that the MI loss in Eq. 1 decreases monotonically.

4 Consistency of Multiple Views

In this section, we present how the consistency of document clustering on target data could be enhanced among multiple views, which is the key issue of our multi-view adaptation method.

According to Lemma 1, minimizing $\phi_D(d, \hat{d})$ for each d can reduce the objective function value iteratively (t denotes round id):

$$\mathcal{C}_D^{(t+1)}(d) = \arg\min_{\hat{d}} \Big[\alpha \cdot \mathcal{D}(p(w|d)||q^{(t)}(w|\hat{d})) + (1-\alpha) \cdot \mathcal{D}(p(l|d)||q^{(t)}(l|\hat{d}))\Big] \quad (2)$$

In each iteration, the optimal document clustering function $\mathcal{C}_D^{(t+1)}$ is to minimize the weighted sum of KL-divergences used in word-view and link-view document clustering functions as shown above. The optimal word-view and link-view clustering functions can be denoted as follows:

$$\mathcal{C}_{D_W}^{(t+1)}(d) = \arg\min_{\hat{d}} \mathcal{D}(p(w|d)||q^{(t)}(w|\hat{d})) \quad (3)$$

$$\mathcal{C}_{D_L}^{(t+1)}(d) = \arg\min_{\hat{d}} \mathcal{D}(p(l|d)||q^{(t)}(l|\hat{d})) \quad (4)$$

Our central idea is that the document clusterings $\mathcal{C}_{D_W}^{(t+1)}$ and $\mathcal{C}_{D_L}^{(t+1)}$ based on the two views are drawn closer in each iteration due to the word and link clusterings that bring together seemingly unrelated source-specific and target-specific features. Meanwhile, $\mathcal{C}_D^{(t+1)}$ combines the two views and reallocates the documents so that it remains consistent with the view-based clusterings as much as possible. The more consistent the views, the better the document clustering, and then the better the word and link clustering, which creates a positive cycle.

4.1 Disagreement Rate of Views

For any document, a consistency indicator function with respect to the two view-based clusterings can be defined as follows (t is omitted for simplicity):

[2]Due to space limit, the proof of all lemmas will be given in a long version of the paper.

Definition 1 (Indicator function) *For any $d \in D$,*

$$\delta_{\mathcal{C}_{D_W}, \mathcal{C}_{D_L}}(d) = \begin{cases} 1, & \text{if } \mathcal{C}_{D_W}(d) = \mathcal{C}_{D_L}(d); \\ 0, & \text{otherwise} \end{cases}$$

Then we define the disagreement rate between two view-based clustering functions:

Definition 2 (Disagreement rate)

$$\eta(\mathcal{C}_{D_W}, \mathcal{C}_{D_L}) = 1 - \frac{\sum_{d \in D} \delta_{\mathcal{C}_{D_W}, \mathcal{C}_{D_L}}(d)}{|D|} \quad (5)$$

Abney (2002) suggests that the disagreement rate of two independent hypotheses upper-bounds the error rate of either hypothesis. By minimizing the disagreement rate on unlabeled data, the error rate of each view can be minimized (so does the overall error). However, Eq. 5 is not continuous nor convex, which is difficult to optimize directly. By using the optimization based on Lemma 1, we can show empirically that disagreement rate is monotonically decreased (see Section 5).

4.2 View Combination

In practice, view-based document clusterings in Eq. 3 and 4 are not computed explicitly. Instead, Eq. 2 directly optimizes view combination and produces the document clustering. Therefore, it is necessary to disclose how consistent it could be with the view-based clusterings.

Suppose $\Omega = \{\mathcal{F}_D | \mathcal{F}_D(d) = \hat{d}, \hat{d} \in \hat{D}\}$ is the set of all document clustering functions. For any $\mathcal{F}_D \in \Omega$, we obtain the disagreement rate $\eta(\mathcal{F}_D, \mathcal{C}_{D_W} \cap \mathcal{C}_{D_L})$, where $\mathcal{C}_{D_W} \cap \mathcal{C}_{D_L}$ denotes the clustering resulting from the overlap of the view-based clusterings.

Lemma 2 *$\mathcal{C}_D$ always minimizes the disagreement rate for any $\mathcal{F}_D \in \Omega$ such that*

$$\eta(\mathcal{C}_D, \mathcal{C}_{D_W} \cap \mathcal{C}_{D_L}) = \min_{\mathcal{F}_D \in \Omega} \eta(\mathcal{F}_D, \mathcal{C}_{D_W} \cap \mathcal{C}_{D_L})$$

Meanwhile, $\eta(\mathcal{C}_D, \mathcal{C}_{D_W} \cap \mathcal{C}_{D_L}) = \eta(\mathcal{C}_{D_W}, \mathcal{C}_{D_L})$.

Lemma 2 suggests that IMAM always finds the document clustering with the minimal disagreement rate to the overlap of view-based clusterings, and the minimal value of disagreement rate equals to the disagreement rate of the view-based clusterings.

Table 1: View disagreement rate η and error rate ϵ that decrease with iterations and their Pearson's correlation γ.

Round		1	2	3	4	5	γ
DA-EC	ϵ	0.194	0.153	0.149	0.144	0.144	0.998
	η	0.340	0.132	0.111	0.101	0.095	
DA-NT	ϵ	0.147	0.083	0.071	0.065	0.064	0.996
	η	0.295	0.100	0.076	0.069	0.064	
DA-OS	ϵ	0.129	0.064	0.052	0.047	0.041	0.998
	η	0.252	0.092	0.068	0.060	0.052	
DA-ML	ϵ	0.166	0.102	0.071	0.065	0.064	0.984
	η	0.306	0.107	0.076	0.062	0.054	
EC-NT	ϵ	0.311	0.250	0.228	0.219	0.217	0.988
	η	0.321	0.137	0.112	0.096	0.089	

Table 2: Comparison of error rate with baselines.

Data	TSVM	Co-Train	CoCC	MVTL-LM	IMAM
DA-EC	0.214	0.230	0.149	0.192	**0.138**
DA-NT	0.114	0.163	0.106	0.108	**0.069**
DA-OS	0.262	0.175	0.075	0.068	**0.039**
DA-ML	0.107	0.171	0.109	0.183	**0.047**
EC-NT	**0.177**	0.296	0.225	0.261	0.192
EC-OS	0.245	0.175	0.137	0.176	**0.074**
EC-ML	**0.168**	0.206	0.203	0.264	0.173
NT-OS	0.396	0.220	0.107	0.288	**0.070**
NT-ML	0.101	0.132	0.054	0.071	**0.032**
OS-ML	0.179	0.128	0.051	0.126	**0.021**
Average	0.196	0.190	0.122	0.174	**0.085**

5 Experiments and Results

Data and Setup

Cora (McCallum et al., 2000) is an online archive of computer science articles. The documents in the archive are categorized into a hierarchical structure. We selected a subset of Cora, which contains 5 top categories and 10 sub-categories. We used a similar way as Dai et al. (2007) to construct our training and test sets. For each set, we chose two top categories, one as positive class and the other as the negative. Different sub-categories were deemed as different domains. The task is defined as top category classification. For example, the dataset denoted as DA-EC consists of source domain: DA_1(+), EC_1(-); and target domain: DA_2(+), EC_2(-).

The classification error rate ϵ is measured as the proportion of misclassified target documents. In order to avoid the infinity values, we applied Laplacian smoothing when computing the KL-divergence. We tuned α, λ and the number of word/link clusters by cross-validation on the training data.

Results and Discussions

Table 1 shows the monotonic decrease of view disagreement rate η and error rate ϵ with the iterations and their Pearson's correlation γ is nearly perfectly positive. This indicates that IMAM gradually improves adaptation by strengthening the view consistency. This is achieved by the reinforcement of word and link clusterings that draw together target- and source-specific features that are originally unrelated but co-occur with the common features.

We compared IMAM with (1) Transductive SVM (TSVM) (Joachims, 1999) using both words and links features; (2) Co-Training (Blum and Mitchell, 1998); (3) CoCC (Dai et al., 2007): Co-clustering-based single-view transfer learner (with text view only); and (4) MVTL-LM (Zhang et al., 2011): Large-margin-based multi-view transfer learner.

Table 2 shows the results. Co-Training performed a little better than TSVM by boosting the confidence of classifiers built on the distinct views in a complementary way. But since Co-Training doesn't consider the distribution gap, it performed clearly worse than CoCC even though CoCC has only one view.

IMAM significantly outperformed CoCC on all the datasets. In average, the error rate of IMAM is 30.3% lower than that of CoCC. This is because IMAM effectively leverages distinct and complementary views. Compared to CoCC, using source training data to improve the view consistency on target data is the key competency of IMAM.

MVTL-LM performed worse than CoCC. It suggests that instance-based approach is not effective when the data of different domains are drawn from different feature spaces. Although MVTL-LM regulates view consistency, it cannot identify the associations between target- and source-specific features that is the key to the success of adaptation especially when domain gap is large and less commonality could be found. In contrast, CoCC and IMAM uses multi-way clustering to find such correlations.

6 Conclusion

We presented a novel feature-level multi-view domain adaptation approach. The thrust is to incorporate distinct views of document features into the information-theoretic co-clustering framework and strengthen the consistency of views on clustering (i.e., classifying) target documents. The improvements over the state-of-the-arts are significant.

References

Steven Abney. 2002. Bootstrapping. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 360-367.

John Blitzer, Mark Dredze and Fernado 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 440-447.

John Blitzer, Sham Kakade and Dean P. Foster. 2011. Domain Adaptation with Coupled Subspaces. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 173-181.

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.

Minmin Chen, Killian Q. Weinberger and John Blitzer. 2011. Co-Training for Domain Adaptation. In *Proceedings of NIPS*, pages 1-9.

Wenyuan Dai, Gui-Rong Xue, Qiang Yang and Yong Yu. 2007. Co-clustering Based Classification for Out-of-domain Documents. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 210-219.

Sanjoy Dasgupta, Michael L. Littman and David McAllester. 2001. PAC Generalization Bounds for Co-Training. In *Proceeding of NIPS*, pages 375-382.

Hal Daumé III and Daniel Marcu. 2006. Domain Adaptation for Statistical Classifiers. *Journal of Artificial Intelligence Research*, 26(2006):101-126.

Inderjit S. Dhillon, Subramanyam Mallela and Dharmendra S. Modha. 2003. Information-Theoretic Co-clustering. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 210-219.

Thorsten Joachims. 1999. Transductive Inference for Text Classification using Support Vector Machines. In *Proceedings of Sixteenth International Conference on Machine Learning*, pages 200-209.

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.

Andrew K. McCallum, Kamal Nigam, Jason Rennie and Kristie Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval*, 3(2):127-163.

Stephan Rüping and Tobias Scheffer. 2005. Learning with Multiple Views. In *Proceedings of ICML Workshop on Learning with Multiple Views*.

Kanoksri Sarinnapakorn and Miroslav Kubat. 2007. Combining Sub-classifiers in Text Categorization: A DST-Based Solution and a Case Study. *IEEE Transactions Knowledge and Data Engineering*, 19(12):1638-1651.

Karthik Sridharan and Sham M. Kakade. 2008. An Information Theoretic Framework for Multi-view Learning. In *Proceedings of the 21st Annual Conference on Computational Learning Theory*, pages 403-414.

Dan Zhang, Jingrui He, Yan Liu, Luo Si and Richard D. Lawrence. 2011. Multi-view Transfer Learning with a Large Margin Approach. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1208-1216.

Efficient Tree-Based Topic Modeling

Yuening Hu
Department of Computer Science
University of Maryland, College Park
ynhu@cs.umd.edu

Jordan Boyd-Graber
iSchool and UMIACS
University of Maryland, College Park
jbg@umiacs.umd.edu

Abstract

Topic modeling with a tree-based prior has been used for a variety of applications because it can encode correlations between words that traditional topic modeling cannot. However, its expressive power comes at the cost of more complicated inference. We extend the SPARSELDA (Yao et al., 2009) inference scheme for latent Dirichlet allocation (LDA) to tree-based topic models. This sampling scheme computes the exact conditional distribution for Gibbs sampling much more quickly than enumerating all possible latent variable assignments. We further improve performance by iteratively refining the sampling distribution only when needed. Experiments show that the proposed techniques dramatically improve the computation time.

1 Introduction

Topic models, exemplified by latent Dirichlet allocation (LDA) (Blei et al., 2003), discover latent themes present in text collections. "Topics" discovered by topic models are multinomial probability distributions over words that evince thematic coherence. Topic models are used in computational biology, computer vision, music, and, of course, text analysis.

One of LDA's virtues is that it is a simple model that assumes a symmetric Dirichlet prior over its word distributions. Recent work argues for structured distributions that constrain clusters (Andrzejewski et al., 2009), span languages (Jagarlamudi and Daumé III, 2010), or incorporate human feedback (Hu et al., 2011) to improve the quality and flexibility of topic modeling. These models all use different tree-based prior distributions (Section 2).

These approaches are appealing because they preserve conjugacy, making inference using Gibbs sampling (Heinrich, 2004) straightforward. While straightforward, inference isn't cheap. Particularly for interactive settings (Hu et al., 2011), efficient inference would improve perceived latency.

SPARSELDA (Yao et al., 2009) is an efficient Gibbs sampling algorithm for LDA based on a refactorization of the conditional topic distribution (reviewed in Section 3). However, it is not directly applicable to tree-based priors. In Section 4, we provide a factorization for tree-based models within a broadly applicable inference framework that empirically improves the efficiency of inference (Section 5).

2 Topic Modeling with Tree-Based Priors

Trees are intuitive methods for encoding human knowledge. Abney and Light (1999) used tree-structured multinomials to model selectional restrictions, which was later put into a Bayesian context for topic modeling (Boyd-Graber et al., 2007). In both cases, the tree came from WordNet (Miller, 1990), but the tree could also come from domain experts (Andrzejewski et al., 2009).

Organizing words in this way induces *correlations* that are mathematically impossible to represent with a symmetric Dirichlet prior. To see how correlations can occur, consider the generative process. Start with a rooted tree structure that contains internal nodes and leaf nodes. This skeleton is a prior that generates K topics. Like vanilla LDA, these topics are distributions over words. Unlike vanilla LDA, their structure correlates words. Internal nodes have a distribution $\pi_{k,i}$ over children, where $\pi_{k,i}$ comes from per-node Dirichlet parameterized by β_i.[1] Each leaf node is associated with a word, and each word must appear in at least (possibly more than) one leaf node.

To generate a word from topic k, start at the root. Select a child $x_0 \sim \text{Mult}(\pi_{k,\text{ROOT}})$, and traverse the tree until reaching a leaf node. Then emit the leaf's associated word. This walk replaces the draw from a topic's multinomial distribution over words.

[1] Choosing these Dirichlet priors specifies the direction (i.e., positive or negative) and strength of correlations that appear.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 275–279,
Jeju, Republic of Korea, 8-14 July 2012.

The rest of the generative process for LDA remains the same, with θ, the per-document topic multinomial, and z, the topic assignment.

This tree structure encodes correlations. The closer types are in the tree, the more correlated they are. Because types can appear in multiple leaf nodes, this encodes polysemy. The path that generates a token is an additional latent variable we must sample.

Gibbs sampling is straightforward because the tree-based prior maintains conjugacy (Andrzejewski et al., 2009). We integrate the per-document topic distributions θ and the transition distributions π. The remaining latent variables are the topic assignment z and path l, which we sample jointly:[2]

$$p(z = k, l = \lambda | Z_-, L_-, w) \propto (\alpha_k + n_{k|d}) \prod_{(i \to j) \in \lambda} \frac{\beta_{i \to j} + n_{i \to j|k}}{\sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})} \quad (1)$$

where $n_{k|d}$ is topic k's count in the document d; α_k is topic k's prior; Z_- and L_- are topic and path assignments excluding $w_{d,n}$; $\beta_{i \to j}$ is the prior for edge $i \to j$, $n_{i \to j|t}$ is the count of edge $i \to j$ in topic k; and j' denotes other children of node i.

The complexity of computing the sampling distribution is $O(KLS)$ for models with K topics, paths at most L nodes long, and at most S paths per word type. In contrast, for vanilla LDA the analogous conditional sampling distribution requires $O(K)$.

3 Efficient LDA

The SPARSELDA (Yao et al., 2009) scheme for speeding inference begins by rearranging LDA's sampling equation into three terms:[3]

$$p(z = k | Z_-, w) \propto (\alpha_k + n_{k|d}) \frac{\beta + n_{w|k}}{\beta V + n_{\cdot|k}} \quad (2)$$
$$\propto \underbrace{\frac{\alpha_k \beta}{\beta V + n_{\cdot|k}}}_{s_{\text{LDA}}} + \underbrace{\frac{n_{k|d} \beta}{\beta V + n_{\cdot|k}}}_{r_{\text{LDA}}} + \underbrace{\frac{(\alpha_k + n_{k|d}) n_{w|k}}{\beta V + n_{\cdot|k}}}_{q_{\text{LDA}}}$$

Following their lead, we call these three terms "buckets". A bucket is the *total* probability mass marginalizing over latent variable assignments (i.e., $s_{\text{LDA}} \equiv \sum_k \frac{\alpha_k \beta}{\beta V + n_{\cdot|k}}$, similarly for the other buckets). The three buckets are a smoothing only bucket s_{LDA}, document topic bucket r_{LDA}, and topic word bucket q_{LDA} (we use the "LDA" subscript to contrast with our method, for which we use the same bucket names without subscripts).

Caching the buckets' total mass speeds the computation of the sampling distribution. Bucket s_{LDA} is shared by all tokens, and bucket r_{LDA} is shared by a document's tokens. Both have simple constant time updates. Bucket q_{LDA} has to be computed specifically for each token, but only for the (typically) few types with non-zero counts in a topic.

To sample from the conditional distribution, first sample *which* bucket you need and then (and only then) select a topic *within* that bucket. Because the topic-term bucket q_{LDA} often has the largest mass and has few non-zero terms, this speeds inference.

4 Efficient Inference in Tree-Based Models

In this section, we extend the sampling techniques for SPARSELDA to tree-based topic modeling. We first factor Equation 1:

$$p(z = k, l = \lambda | Z_-, L_-, w) \propto (\alpha_k + n_{k|d}) N_{k,\lambda}^{-1} [S_\lambda + O_{k,\lambda}]. \quad (3)$$

Henceforth we call $N_{k,\lambda}$ the normalizer for path λ in topic k, S_λ the smoothing factor for path λ, and $O_{k,\lambda}$ the observation for path λ in topic k, which are

$$N_{k,\lambda} = \prod_{(i \to j) \in \lambda} \sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})$$
$$S_\lambda = \prod_{(i \to j) \in \lambda} \beta_{i \to j} \quad (4)$$
$$O_{k,\lambda} = \prod_{(i \to j) \in \lambda} (\beta_{i \to j} + n_{i \to j|k}) - \prod_{(i \to j) \in \lambda} \beta_{i \to j}.$$

Equation 3 can be rearranged in the same way as Equation 5, yielding buckets analogous to SPARSELDA's,

$$p(z = k, l = \lambda | Z_-, L_-, w) \propto \underbrace{\frac{\alpha_k S_\lambda}{N_{k,\lambda}}}_{s} + \underbrace{\frac{n_{k|d} S_\lambda}{N_{k,\lambda}}}_{r} + \underbrace{\frac{(\alpha_k + n_{k|d}) O_{k,\lambda}}{N_{k,\lambda}}}_{q}. \quad (5)$$

Buckets sum both topics and paths. The sampling process is much the same as for SPARSELDA: select *which* bucket and then select a topic / path combination *within* the bucket (for a slightly more complex example, see Algorithm 1).

[2] For clarity, we omit indicators that ensure λ ends at $w_{d,n}$.

[3] To ease notation we drop the $_{d,n}$ subscript for z and w in this and future equations.

Recall that one of the benefits of SPARSELDA was that s was shared across tokens. This is no longer possible, as $N_{k,\lambda}$ is distinct for each path in tree-based LDA. Moreover, $N_{k,\lambda}$ is coupled; changing $n_{i \to j|k}$ in one path changes the normalizers of all cousin paths (paths that share some node i).

This negates the benefit of caching s, but we recover some of the benefits by splitting the normalizer to two parts: the "root" normalizer from the root node (shared by all paths) and the "downstream" normalizer. We precompute which paths share downstream normalizers; all paths are partitioned into cousin sets, defined as sets for which changing the count of one member of the set changes the downstream normalizer of other paths in the set. Thus, when updating the counts for path l, we only recompute $N_{k,l'}$ for all l' in the cousin set.

SPARSELDA's computation of q, the topic-word bucket, benefits from topics with unobserved (i.e., zero count) types. In our case, any non-zero path, a path with *any* non-zero edge, contributes.[4] To quickly determine whether a path contributes, we introduce an **edge-masked count** (EMC) for each path. Higher order bits encode whether edges have been observed and lower order bits encode the number of times the path has been observed. For example, if a path of length three only has its first two edges observed, its EMC is $\overline{110}00000$. If the same path were observed seven times, its EMC is $\overline{111}00111$. With this formulation we can ignore any paths with a zero EMC.

Efficient sampling with refined bucket While caching the sampling equation as described in the previous section improved the efficiency, the smoothing only bucket s is small, but computing the associated mass is costly because it requires us to consider all topics and paths. This is not a problem for *SparseLDA* because s is shared across all tokens. However, we can achieve computational gains with an upper bound on s,

$$s = \sum_{k,\lambda} \frac{\alpha_k \prod_{(i \to j) \in \lambda} \beta_{i \to j}}{\prod_{(i \to j) \in \lambda} \sum_{j'} (\beta_{i \to j'} + n_{i \to j'|k})}$$
$$\leq \sum_{k,\lambda} \frac{\alpha_k \prod_{(i \to j) \in \lambda} \beta_{i \to j}}{\prod_{(i \to j) \in \lambda} \sum_{j'} \beta_{i \to j'}} = s'. \quad (6)$$

A sampling algorithm can take advantage of this by not explicitly calculating s. Instead, we use s' as proxy, and only compute the exact s if we hit the bucket s' (Algorithm 1). Removing s' and always computing s yields the first algorithm in Section 4.

Algorithm 1 SAMPLING WITH REFINED BUCKET

```
for word w in this document do
    sample = rand() *(s' + r + q)
    if sample < s' then
        compute s
        sample = sample *(s + r + q)/(s' + r + q)
        if sample < s then
            return topic k and path λ sampled from s
        sample -= s
    else
        sample -= s'
    if sample < r then
        return topic k and path λ sampled from r
    sample -= r
    return topic k and path λ sampled from q
```

Sorting Thus far, we described techniques for efficiently computing buckets, but quickly sampling assignments within a bucket is also important. Here we propose two techniques to consider latent variable assignments in *decreasing* order of probability mass. By considering fewer possible assignments, we can speed sampling at the cost of the overhead of maintaining sorted data structures. We sort topics' prominence within a document (SD) and sort the topics and paths of a word (SW).

Sorting topics' prominence within a document (SD) can improve sampling from r and q; when we need to sample within a bucket, we consider paths in decreasing order of $n_{k|d}$.

Sorting path prominence for a word (SW) can improve our ability to sample from q. The edge-masked count (EMC), as described above, serves as a proxy for the probability of a path and topic. If, when sampling a topic and path from q, we sample based on the decreasing EMC, which roughly correlates with path probability.

5 Experiments

In this section, we compare the running time[5] of our sampling algorithm (FAST) and our algorithm with the refined bucket (RB) against the unfactored Gibbs sampler (NAÏVE) and examine the effect of sorting.

Our corpus has editorials from New York Times

[4] C.f. observed paths, where *all* edges are non-zero.

[5] Mean of five chains on a 6-Core 2.8-GHz CPU, 16GB RAM

Number of Topics				
	T50	T100	T200	T500
NAIVE	5.700	12.655	29.200	71.223
FAST	4.935	9.222	17.559	40.691
FAST-RB	2.937	4.037	5.880	8.551
FAST-RB-SD	2.675	3.795	5.400	8.363
FAST-RB-SW	2.449	3.363	4.894	7.404
FAST-RB-SDW	2.225	3.241	4.672	7.424
Vocabulary Size				
	V5000	V10000	V20000	V30000
NAÏVE	4.815	12.351	28.783	51.088
FAST	2.897	9.063	20.460	38.119
FAST-RB	1.012	3.900	9.777	20.040
FAST-RB-SD	0.972	3.684	9.287	18.685
FAST-RB-SW	0.889	3.376	8.406	16.640
FAST-RB-SDW	0.828	3.113	7.777	15.397
Number of Correlations				
	C50	C100	C200	C500
NAÏVE	11.166	12.586	13.000	15.377
FAST	8.889	9.165	9.177	8.079
FAST-RB	3.995	4.078	3.858	3.156
FAST-RB-SD	3.660	3.795	3.593	3.065
FAST-RB-SW	3.272	3.363	3.308	2.787
FAST-RB-SDW	3.026	3.241	3.091	2.627

Table 1: The average running time per iteration (S) over 100 iterations, averaged over 5 seeds. Experiments begin with 100 topics, 100 correlations, vocab size 10000 and then vary one dimension: number of topics (top), vocabulary size (middle), and number of correlations (bottom).

from 1987 to 1996.[6] Since we are interested in varying vocabulary size, we rank types by average tf-idf and choose the top V. WordNet 3.0 generates the correlations between types. For each synset in WordNet, we generate a subtree with all types in the synset—that are also in our vocabulary—as leaves connected to a common parent. This subtree's common parent is then attached to the root node.

We compared the FAST and FAST-RB against NAÏVE (Table 1) on different numbers of topics, various vocabulary sizes and different numbers of correlations. FAST is consistently faster than NAÏVE and FAST-RB is consistently faster than FAST. Their benefits are clearer as distributions become sparse (e.g., the first iteration for FAST is slower than later iterations). Gains accumulate as the topic number increases, but decrease a little with the vocabulary size. While both sorting strategies reduce time, sorting topics and paths for a word (SW) helps more than sorting topics in a document (SD), and combining the two is (with one exception) better than either alone.

[6] 13284 documents, 41554 types, and 2714634 tokens.

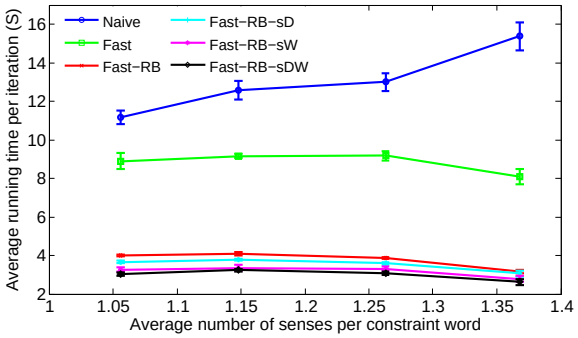


Figure 1: The average running time per iteration against the average number of senses per correlated words.

As more correlations are added, NAÏVE's time increases while that of FAST-RB decreases. This is because the number of non-zero paths for uncorrelated words decreases as more correlations are added to the model. Since our techniques save computation for every zero path, the overall computation decreases as correlations push uncorrelated words to a limited number of topics (Figure 1). Qualitatively, when the synset with "king" and "baron" is added to a model, it is associated with "drug, inmate, colombia, waterfront, baron" in a topic; when "king" is correlated with "queen", the associated topic has "king, parade, museum, queen, jackson" as its most probable words. These represent reasonable disambiguations. In contrast to previous approaches, inference speeds up as topics become more semantically coherent (Boyd-Graber et al., 2007).

6 Conclusion

We demonstrated efficient inference techniques for topic models with tree-based priors. These methods scale well, allowing for faster exploration of models that use semantics to encode correlations without sacrificing accuracy. Improved scalability for such algorithms, especially in distributed environments (Smola and Narayanamurthy, 2010), could improve applications such as cross-language information retrieval, unsupervised word sense disambiguation, and knowledge discovery via interactive topic modeling.

Acknowledgments

We would like to thank David Mimno and the anonymous reviewers for their helpful comments. This work was supported by the Army Research Laboratory through ARL Cooperative Agreement W911NF-09-2-0072. Any opinions or conclusions expressed are the authors' and do not necessarily reflect those of the sponsors.

References

Steven Abney and Marc Light. 1999. Hiding a semantic hierarchy in a Markov model. In *Proceedings of the Workshop on Unsupervised Learning in Natural Language Processing*.

David Andrzejewski, Xiaojin Zhu, and Mark Craven. 2009. Incorporating domain knowledge into topic modeling via Dirichlet forest priors. In *Proceedings of International Conference of Machine Learning*.

David M. Blei, Andrew Ng, and Michael Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Jordan Boyd-Graber, David M. Blei, and Xiaojin Zhu. 2007. A topic model for word sense disambiguation. In *Proceedings of Emperical Methods in Natural Language Processing*.

Gregor Heinrich. 2004. Parameter estimation for text analysis. Technical report. http://www.arbylon.net/publications/text-est.pdf.

Yuening Hu, Jordan Boyd-Graber, and Brianna Satinoff. 2011. Interactive topic modeling. In *Association for Computational Linguistics*.

Jagadeesh Jagarlamudi and Hal Daumé III. 2010. Extracting multilingual topics from unaligned corpora. In *Proceedings of the European Conference on Information Retrieval (ECIR)*.

George A. Miller. 1990. Nouns in WordNet: A lexical inheritance system. *International Journal of Lexicography*, 3(4):245–264.

Alexander J. Smola and Shravan Narayanamurthy. 2010. An architecture for parallel topic models. *International Conference on Very Large Databases*, 3.

Limin Yao, David Mimno, and Andrew McCallum. 2009. Efficient methods for topic model inference on streaming document collections. In *Knowledge Discovery and Data Mining*.

Learning Better Rule Extraction with Translation Span Alignment

Jingbo Zhu Tong Xiao Chunliang Zhang
Natural Language Processing Laboratory
Northeastern University, Shenyang, China
{zhujingbo,xiaotong,zhangcl}@mail.neu.edu.cn

Abstract

This paper presents an unsupervised approach to learning *translation span alignments* from parallel data that improves syntactic rule extraction by deleting spurious word alignment links and adding new valuable links based on bilingual translation span correspondences. Experiments on Chinese-English translation demonstrate improvements over standard methods for tree-to-string and tree-to-tree translation.

1 Introduction

Most syntax-based statistical machine translation (SMT) systems typically utilize word alignments and parse trees on the source/target side to learn syntactic transformation rules from parallel data. The approach suffers from a practical problem that even one spurious (word alignment) link can prevent some desirable syntactic translation rules from extraction, which can in turn affect the quality of translation rules and translation performance (May and Knight 2007; Fossum *et al.* 2008). To address this challenge, a considerable amount of previous research has been done to improve alignment quality by incorporating some statistics and linguistic heuristics or syntactic information into word alignments (Cherry and Lin 2006; DeNero and Klein 2007; May and Knight 2007; Fossum *et al.* 2008; Hermjakob 2009; Liu *et al.* 2010).

Unlike their efforts, this paper presents a simple approach that automatically builds the *translation span alignment* (TSA) of a sentence pair by utilizing a phrase-based forced decoding technique, and then improves syntactic rule extraction by deleting spurious links and adding new valuable links based on bilingual translation span correspondences. The proposed approach has two promising properties.

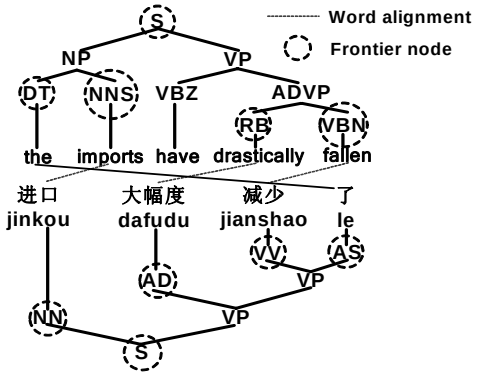


Figure 1. A real example of Chinese-English sentence pair with word alignment and both-side parse trees.

Some blocked Tree-to-string Rules:
r_1: AS(了) → have
r_2: NN(进口) → the imports
r_3: S (NN:x_1 VP:x_2) → x_1 x_2
Some blocked Tree-to-tree Rules:
r_4: AS(了) → VBZ(have)
r_5: NN(进口) → NP(DT(the) NNS(imports))
r_6: S(NN:x_1 VP:x_2) → S(NP:x_1 VP:x_2)
r_7: VP(AD:x_1 VP(VV:x_2 AS:x_3)) → VP(VBZ:x_3 ADVP(RB:x_1 VBN:x_2))

Table 1. Some useful syntactic rules are blocked due to the spurious link between "了" and "*the*".

Firstly, The TSAs are constructed in an unsupervised learning manner, and optimized by the translation model during the forced decoding process, without using any statistics and linguistic heuristics or syntactic constraints. Secondly, our approach is independent of the word alignment-based algorithm used to extract translation rules, and easy to implement.

2 Translation Span Alignment Model

Different from word alignment, TSA is a process of identifying span-to-span alignments between parallel sentences. For each translation span pair,

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 280–284,
Jeju, Republic of Korea, 8-14 July 2012.

1. Extract phrase translation rules R from the parallel corpus with word alignment, and construct a phrase-based translation model M.
2. Apply M to implement phrase-based forced decoding on each training sentence pair (c, e), and output its best derivation d^* that can transform c into e.
3. Build a TSA of each sentence pair (c, e) from its best derivation d^*, in which each rule r in d^* is used to form a translation span pair $\{src(r)<=>tgt(r)\}$.

Figure 2. TSA generation algorithm. *src(r)* and *tgt(r)* indicate the source and target side of rule *r*.

its source (or target) span is a sequence of source (or target) words. Given a source sentence $c=c_1...c_n$, a target sentence $e=e_1...e_m$, and its word alignment A, a translation span pair τ is a pair of source span $(c_i...c_j)$ and target span $(e_p...e_q)$

$$\tau = (c_i^j \Leftrightarrow e_p^q)$$

where τ indicates that the source span $(c_i...c_j)$ and the target span $(e_p...e_q)$ are translational equivalent. We do not require that τ must be consistent with the associated word alignment A in a TSA model.

Figure 2 depicts the TSA generation algorithm in which a *phrase-based forced decoding* technique is adopted to produce the TSA of each sentence pair. In this work, we do not apply syntax-based forced decoding (e.g., tree-to-string) because phrase-based models can achieve the state-of-the-art translation quality with a large amount of training data, and are not limited by any constituent boundary based constraints for decoding.

Formally, given a sentence pair (c, e), the phrase-based forced decoding technique aims to search for the *best* derivation d^* among all consistent derivations that convert the given source sentence c into the given target sentence e with respect to the current translation model induced from the training data, which can be expressed by

$$d^* = \arg\max_{d \in D(c,e) \wedge TGT(d)=e} \Pr_\theta(TGT(d) \mid c) \qquad (1)$$

where $D(c,e)$ is the set of candidate derivations that transform c to e, and TGT(d) is a function that outputs the yield of a derivation d. θ indicates parameters of the phrase-based translation model learned from the parallel corpus.

The best derivation d^* produced by forced decoding can be viewed as a sequence of translation steps (i.e., phrase translation rules), expressed by

$$d^* = r_1 \oplus r_2 \oplus ... \oplus r_k ,$$

c = 进口 大幅度 减少 了
e = the imports have drastically fallen
The best derivation d^* produced by forced decoding:
r_1: 进口 → the imports
r_2: 大幅度 减少 → drastically fallen
r_3: 了 → have
Generating TSA from d^*:
[进口]<=>[the imports]
[大幅度 减少]<=>[drastically fallen]
[了]<=>[have]

Table 2. Forced decoding based TSA generation on the example sentence pair in Fig. 1.

where r_i indicates a phrase rule used to form d^*. $\oplus$ is a composition operation that combines rules $\{r_1...r_k\}$ together to produce the target translation.

As mentioned above, the best derivation d^* respects the input sentence pair (c, e). It means that for each phrase translation rule r_i used by d^*, its source (or target) side exactly matches a span of the given source (or target) sentence. The source side $src(r_i)$ and the target side $tgt(r_i)$ of each phrase translation rule r_i in d^* form a translation span pair $\{src(r_i)<=>tgt(r_i)\}$ of (c,e). In other words, the TSA of (c,e) is a set of translation span pairs generated from phrase translation rules used by the best derivation d^*. The forced decoding based TSA generation on the example sentence pair in Figure 1 can be shown in Table 2.

3 Better Rule Extraction with TSAs

To better understand the particular task that we will address in this section, we first introduce a definition of *inconsistent with a translation span alignment*. Given a sentence pair (c, e) with the word alignment A and the translation span alignment P, we call a link $(c_i, e_j) \in A$ *inconsistent* with P, if c_i and e_j are covered respectively by two different translation span pairs in P and vice versa.

$(c_i, e_j) \in A$ inconsistent with $P \Leftrightarrow$

$$\exists \tau \in P : c_i \in src(\tau) \wedge e_j \notin tgt(\tau)$$

$$OR \ \exists \tau \in P : c_i \notin src(\tau) \wedge e_j \in tgt(\tau)$$

where $src(\tau)$ and $tgt(\tau)$ indicate the source and target span of a translation span pair τ.

By this, we will say that a link $(c_i, e_j) \in A$ is a spurious link if it is inconsistent with the given TSA. Table 3 shows that an original link (4→1) are covered by two different translation span pairs

Source	Target	WA	TSA
1: 进口	1: the	1→2	[1,1]<=>[1,2]
2: 大幅度	2: imports	2→4	[2,3]<=>[4,5]
3: 减少	3: have	3→5	[4,4]<=>[3,3]
4: 了	4: drastically	4→1	
	5: fallen	(null)→3	

Table 3. A sentence pair with the original word alignment (WA) and the translation span alignment (TSA).

([4,4]<=>[3,3]) and ([1,1] <=>[1,2]), respectively. In such a case, we think that this link (4→1) is a spurious link according to this TSA, and should be removed for rule extraction.

Given a resulting TSA *P*, there are four different types of translation span pairs, such as one-to-one, one-to-many, many-to-one, and many-to-many cases. For example, the TSA shown in Table 3 contains a one-to-one span pair ([4,4]<=>[3,3]), a one-to-many span pair ([1,1]<=>[1,2]) and a many-many span pair ([2,3]<=>[4,5]). In such a case, we can learn a confident link from a one-to-one translation span pair that is preferred by the translation model in the forced decoding based TSA generation approach. If such a confident link does not exist in the original word alignment, we consider it as a new valuable link.

Until now, a natural way is to use TSAs to directly improve word alignment quality by deleting some spurious links and adding some new confident links, which in turn improves rule quality and translation quality. In other words, if a desirable translation rule was blocked due to some spurious links, we will output this translation rule. Let's revisit the example in Figure 1 again. The blocked tree-to-string r_3 can be extracted successfully after deleting the spurious link (了, *the*), and a new tree-to-string rule r_1 can be extracted after adding a new confident link (了, *have*) that is inferred from a one-to-one translation span pair [4,4]<=>[3,3].

4 Experiments

4.1 Setup

We utilized a state-of-the-art open-source SMT system NiuTrans (Xiao et al. 2012) to implement syntax-based models in the following experiments. We begin with a training parallel corpus of Chinese-English bitexts that consists of 8.8M Chinese words and 10.1M English words in 350K sentence pairs. The GIZA++ tool was used to perform the bi-directional word alignment between the source and the target sentences, referred to as the *baseline* method. For syntactic translation rule extraction, minimal GHKM (Galley *et al.*, 2004) rules are first extracted from the bilingual corpus whose source and target sides are parsed using the Berkeley parser (Petrov *et al.* 2006). The composed rules are then generated by composing two or three minimal rules. A 5-gram language model was trained on the Xinhua portion of English Gigaword corpus. Beam search and cube pruning techniques (Huang and Chiang 2007) were used to prune the search space for all the systems. The base feature set used for all systems is similar to that used in (Marcu *et al.* 2006), including 14 base features in total such as 5-gram language model, bidirectional lexical and phrase-based translation probabilities. All features were log-linearly combined and their weights were optimized by performing minimum error rate training (MERT) (Och 2003). The development data set used for weight training comes from NIST MT03 evaluation set, consisting of 326 sentence pairs of less than 20 words in each Chinese sentence. Two test sets are NIST MT04 (1788 sentence pairs) and MT05 (1082 sentence pairs) evaluation sets. The translation quality is evaluated in terms of the case-insensitive IBM-BLEU4 metric.

Method	Prec%	Rec%	F1%	Del/Sent	Add/Sent
Baseline	83.07	75.75	79.25	-	-
TSA	84.01	75.46	79.51	1.5	1.1

Table 4. Word alignment precision, recall and F1-score of various methods on 200 sentence pairs of Chinese-English data.

4.2 Effect on Word Alignment

To investigate the effect of the TSA method on word alignment, we designed an experiment to evaluate alignment quality against gold standard annotations. There are 200 random chosen and manually aligned Chinese-English sentence pairs used to assert the word alignment quality. For word alignment evaluation, we calculated precision, recall and F1-score over gold word alignment.

Table 4 depicts word alignment performance of the baseline and TSA methods. We apply the TSAs to refine the baseline word alignments, involving spurious link deletion and new link insertion operations. Table 4 shows our method can yield improvements on precision and F1-score, only causing a little negative effect on recall.

4.3 Translation Quality

Method	# of Rules	MT03	MT04	MT05
Baseline (T2S)	33,769,071	34.10	32.55	30.15
TSA (T2S)	32,652,261	34.61^{+} (+0.51)	33.01^{+} (+0.46)	30.66^{+} (+0.51)
Baseline (T2T)	24,287,206	34.51	32.20	31.78
TSA (T2T)	24,119,719	34.85 (+0.34)	32.92* (+0.72)	32.22^{+} (+0.44)

Table 5. Rule sizes and IBM-BLEU4 (%) scores of baseline and our method (TSA) in tree-to-string (T2S) and tree-to-tree (T2T) translation on Dev set (MT03) and two test sets (MT04 and MT05). + and * indicate significantly better on performance comparison at $p<.05$ and $p<.01$, respectively.

Table 5 depicts effectiveness of our TSA method on translation quality in tree-to-string and tree-to-tree translation tasks. Table 5 shows that our TSA method can improve both syntax-based translation systems. As mentioned before, the resulting TSAs are essentially optimized by the translation model. Based on such TSAs, experiments show that spurious link deletion and new valuable link insertion can improve translation quality for tree-to-string and tree-to-tree systems.

5 Related Work

Previous studies have made great efforts to incorporate statistics and linguistic heuristics or syntactic information into word alignments (Ittycheriah and Roukos 2005; Taskar *et al*. 2005; Moore *et al.* 2006; Cherry and Lin 2006; DeNero and Klein 2007; May and Knight 2007; Fossum *et al.* 2008; Hermjakob 2009; Liu *et al.* 2010). For example, Fossum *et al.* (2008) used a discriminatively trained model to identify and delete incorrect links from original word alignments to improve string-to-tree transformation rule extraction, which incorporates four types of features such as lexical and syntactic features. This paper presents an approach to incorporating translation span alignments into word alignments to delete spurious links and add new valuable links.

Some previous work directly models the syntactic correspondence in the training data for syntactic rule extraction (Imamura 2001; Groves *et al.* 2004; Tinsley *et al.* 2007; Sun *et al.* 2010a, 2010b; Pauls *et al.* 2010). Some previous methods infer syntactic correspondences between the source and the target languages through word alignments and constituent boundary based syntactic constraints. Such a syntactic alignment method is sensitive to word alignment behavior. To combat this, Pauls *et al*. (2010) presented an unsupervised ITG alignment model that directly aligns syntactic structures for string-to-tree transformation rule extraction. One major problem with syntactic structure alignment is that syntactic divergence between languages can prevent accurate syntactic alignments between the source and target languages.

May and Knight (2007) presented a syntactic re-alignment model for syntax-based MT that uses syntactic constraints to re-align a parallel corpus with word alignments. The motivation behind their methods is similar to ours. Our work differs from (May and Knight 2007) in two major respects. First, the approach proposed by May and Knight (2007) first utilizes the EM algorithm to obtain Viterbi derivation trees from derivation forests of each (tree, string) pair, and then produces Viterbi alignments based on obtained derivation trees. Our forced decoding based approach searches for the best derivation to produce translation span alignments that are used to improve the extraction of translation rules. Translation span alignments are optimized by the translation model. Secondly, their models are only applicable for syntax-based systems while our method can be applied to both phrase-based and syntax-based translation tasks.

6 Conclusion

This paper presents an unsupervised approach to improving syntactic transformation rule extraction by deleting spurious links and adding new valuable links with the help of bilingual translation span alignments that are built by using a phrase-based forced decoding technique. In our future work, it is worth studying how to combine the best of our approach and discriminative word alignment models to improve rule extraction for SMT models.

Acknowledgments

This research was supported in part by the National Science Foundation of China (61073140), the Specialized Research Fund for the Doctoral Program of Higher Education (20100042110031) and the Fundamental Research Funds for the Central Universities in China.

References

Colin Cherry and Dekang Lin. 2006. Soft syntactic constraints for word alignment through discriminative training. In *Proc. of ACL*.

John DeNero and Dan Klein. 2007. Tailoring word alignments to syntactic machine translation. In *Proc. of ACL*.

Victoria Fossum, Kevin Knight and Steven Abney. 2008. Using syntax to improve word alignment precision for syntax-based machine translation. In *Proc. of the Third Workshop on Statistical Machine Translation*, pages 44-52.

Michel Galley, Mark Hopkins, Kevin Knight and Daniel Marcu. 2004. What's in a translation rule? In *Proc. of HLT-NAACL 2004*, pp273-280.

Declan Groves, Mary Hearne and Andy Way. 2004. Robust sub-sentential alignment of phrase-structure trees. In *Proc. of COLING,* pp1072-1078.

Ulf Hermjakob. 2009. Improved word alignment with statistics and linguistic heuristics. In *Proc. of EMNLP*, pp229-237

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proc. of ACL*, pp144-151.

Kenji Imamura. 2001. Hierarchical Phrase Alignment Harmonized with Parsing. In *Proc. of NLPRS*, pp377-384.

Abraham Ittycheriah and Salim Roukos. 2005. A maximum entropy word aligner for Arabic-English machine translation. In *Proc. of HLT/EMNLP*.

Yang Liu, Qun Liu and Shouxun Lin. 2010. Discriminative word alignment by linear modeling. *Computational Linguistics*, 36(3):303-339

Daniel Marcu, Wei Wang, Abdessamad Echihabi and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proc. of EMNLP*, pp44-52.

Jonathan May and Kevin Knight. 2007. Syntactic realignment models for machine translation. In *Proc. of EMNLP-CoNLL*.

Robert C. Moore, Wen-tau Yih and Andreas Bode. 2006. Improved discriminative bilingual word alignment. In *Proc. of ACL*

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL*.

Adam Pauls, Dan Klein, David Chiang and Kevin Knight. 2010. Unsupervised syntactic alignment with inversion transduction grammars. In *Proc. of NAACL*, pp118-126

Slav Petrov, Leon Barrett, Roman Thibaux and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. of ACL*, pp433-440.

Jun Sun, Min Zhang and Chew Lim Tan. 2010a. Exploring Syntactic Structural Features for Sub-Tree Alignment Using Bilingual Tree Kernels. In *Proc. of ACL*, pp306-315.

Jun Sun, Min Zhang and Chew Lim Tan. 2010b. Discriminative Induction of Sub-Tree Alignment using Limited Labeled Data. In *Proc. of COLING*, pp1047-1055.

Ben Taskar, Simon Lacoste-Julien and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proc. of HLT/EMNLP*

John Tinsley, Ventsislav Zhechev, Mary Hearne and Andy Way. 2007. Robust language pair-independent sub-tree alignment. In *Proc. of MT Summit XI*.

Tong Xiao, Jingbo Zhu, Hao Zhang and Qiang Li. 2012. NiuTrans: An Open Source Toolkit for Phrase-based and Syntax-based Machine Translation. In *Proceedings of ACL*, demonstration session

Enhancing Statistical Machine Translation with Character Alignment

Ning Xi, Guangchao Tang, Xinyu Dai, Shujian Huang, Jiajun Chen
State Key Laboratory for Novel Software Technology,
Department of Computer Science and Technology,
Nanjing University, Nanjing, 210046, China
{xin,tanggc,dxy,huangsj,chenjj}@nlp.nju.edu.cn

Abstract

The dominant practice of statistical machine translation (SMT) uses the same Chinese word segmentation specification in both alignment and translation rule induction steps in building Chinese-English SMT system, which may suffer from a suboptimal problem that word segmentation better for alignment is not necessarily better for translation. To tackle this, we propose a framework that uses two different segmentation specifications for alignment and translation respectively: we use Chinese character as the basic unit for alignment, and then convert this alignment to conventional word alignment for translation rule induction. Experimentally, our approach outperformed two baselines: fully word-based system (using word for both alignment and translation) and fully character-based system, in terms of alignment quality and translation performance.

1 Introduction

Chinese Word segmentation is a necessary step in Chinese-English statistical machine translation (SMT) because Chinese sentences do not delimit words by spaces. The key characteristic of a Chinese word segmenter is the segmentation specification[1]. As depicted in Figure 1(a), the dominant practice of SMT uses the same word segmentation for both word alignment and translation rule induction. For brevity, we will refer to the word segmentation of the bilingual corpus as *word segmentation for alignment* (WSA for short), because it determines the basic tokens for alignment; and refer to the word segmentation of the aligned corpus as *word segmentation for rules* (WSR for short), because it determines the basic tokens of translation rules[2], which also determines how the translation rules would be matched by the source sentences.

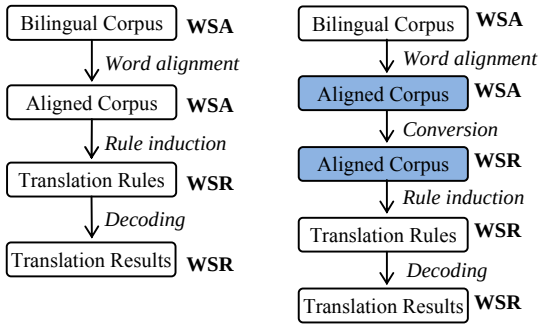


Figure 1. WSA and WSR in SMT pipeline

It is widely accepted that word segmentation with a higher F-score will not necessarily yield better translation performance (Chang et al., 2008; Zhang et al., 2008; Xiao et al., 2010). Therefore, many approaches have been proposed to learn word segmentation suitable for SMT. These approaches were either complicated (Ma et al., 2007; Chang et al., 2008; Ma and Way, 2009; Paul et al., 2010), or of high computational complexity (Chung and Gildea 2009; Duan et al., 2010). Moreover, they implicitly assumed that WSA and WSR should be equal. This requirement may lead to a suboptimal problem that word segmentation better for alignment is not necessarily better for translation.

To tackle this, we propose a framework that uses different word segmentation specifications as WSA and WSR respectively, as shown Figure 1(b). We investigate a solution in this framework: first, we use Chinese character as the basic unit for alignment, viz. *character alignment*; second, we use a simple method (Elming and Habash, 2007) to convert the character alignment to conventional word alignment for translation rule induction. In the

[1] We hereafter use "word segmentation" for short.

[2] Interestingly, word is also a basic token in syntax-based rules.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 285–290,
Jeju, Republic of Korea, 8-14 July 2012.

experiment, our approach consistently outperformed two baselines with three different word segmenters: fully word-based system (using word for both alignment and translation) and fully character-based system, in terms of alignment quality and translation performance.

The remainder of this paper is structured as follows: Section 2 analyzes the influences of WSA and WSR on SMT respectively; Section 3 discusses how to convert character alignment to word alignment; Section 4 presents experimental results, followed by conclusions and future work in section 5.

2 Understanding WSA and WSR

We propose a solution to tackle the suboptimal problem: using Chinese character for alignment while using Chinese word for translation. Character alignment differs from conventional word alignment in the basic tokens of the Chinese side of the training corpus[3]. Table 1 compares the token distributions of character-based corpus (*CCorpus*) and word-based corpus (*WCorpus*). We see that the *WCorpus* has a longer-tailed distribution than the *CCorpus*. More than 70% of the unique tokens appear less than 5 times in *WCorpus*. However, over half of the tokens appear more than or equal to 5 times in the *CCorpus*. This indicates that modeling word alignment could suffer more from data sparsity than modeling character alignment.

Table 2 shows the numbers of the unique tokens (*#UT*) and unique bilingual token pairs (*#UTP*) of the two corpora. Consider two extensively features, fertility and translation features, which are extensively used by many state-of-the-art word aligners. The number of parameters w.r.t. fertility features grows linearly with *#UT* while the number of parameters w.r.t. translation features grows linearly with *#UTP*. We compare *#UT* and *#UTP* of both corpora in Table 2. As can be seen, *CCorpus* has less *UT* and *UTP* than *WCorpus*, i.e. character alignment model has a compact parameterization than word alignment model, where the compactness of parameterization is shown very important in statistical modeling (Collins, 1999).

Another advantage of character alignment is the reduction in alignment errors caused by word segmentation errors. For example, "切尼 (Cheney)" and "愿 (will)" are wrongly merged into one word 切尼愿 by the word segmenter, and 切尼愿 wrongly aligns to a comma in English sentence in the word alignment; However, both 切 and 尼 align to "*Cheney*" correctly in the character alignment. However, this kind of errors cannot be fixed by methods which learn new words by packing already segmented words, such as word packing (Ma et al., 2007) and Pseudo-word (Duan et al., 2010).

Frequency	Characters (%)	Words (%)
1	27.22	45.39
2	11.13	14.61
3	6.18	6.47
4	4.26	4.32
5(+)	50.21	29.21

Table 1 Token distribution of *CCorpus* and *WCorpus*

Stats.	Characters	Words
#UT	9.7K	88.1K
#UTP	15.8M	24.2M

Table 2 *#UT* and *#UTP* in *CCorpus* and *WCorpus*

As character could preserve more meanings than word in Chinese, it seems that a character can be wrongly aligned to many English words by the aligner. However, we found this can be avoided to a great extent by the basic features (co-occurrence and distortion) used by many alignment models. For example, we observed that the four characters of the non-compositional word "阿拉法特 (Arafat)" align to *Arafat* correctly, although these characters preserve different meanings from that of *Arafat*. This can be attributed to the frequent co-occurrence (192 times) of these characters and *Arafat* in *CCorpus*. Moreover,法 usually means *France* in Chinese, thus it may co-occur very often with *France* in *CCorpus*. If both *France* and *Arafat* appear in the English sentence, 法 may wrongly align to *France*. However, if 阿 aligns to *Arafat*, 法 will probably align to *Arafat*, because aligning 法 to *Arafat* could result in a lower distortion cost than aligning it to *France*.

Different from alignment, translation is a pattern matching procedure (Lopez, 2008). WSR determines how the translation rules would be matched by the source sentences. For example, if we use translation rules with character as WSR to translate name entities such as the non-compositional word 阿拉法特, i.e. translating literally, we may get a wrong translation. That's because the linguistic

[3] Several works have proposed to use character (letter) on both sides of the parallel corpus for SMT between similar (European) languages (Vilar et al., 2007; Tiedemann, 2009), however, Chinese is not similar to English.

knowledge that the four characters convey a specific meaning different from the characters has been lost, which cannot always be totally recovered even by using phrase in phrase-based SMT systems (see Chang et al. (2008) for detail). Duan et al. (2010) and Paul et al., (2010) further pointed out that coarser-grained segmentation of the source sentence do help capture more contexts in translation. Therefore, rather than using character, using coarser-grained, at least as coarser as the conventional word, as WSR is quite necessary.

3 Converting Character Alignment to Word Alignment

In order to use word as WSR, we employ the same method as Elming and Habash (2007)[4] to convert the character alignment (*CA*) to its word-based version (*CA'*) for translation rule induction. The conversion is very intuitive: for every English-Chinese word pair (e, c) in the sentence pair, we align c to e as a link in *CA'*, if and only if there is at least one Chinese character of c aligns to e in *CA*.

Given two different segmentations A and B of the same sentence, it is easy to prove that if every word in A is finer-grained than the word of B at the corresponding position, the conversion is unambiguity (we omit the proof due to space limitation). As character is a finer-grained than its original word, character alignment can always be converted to alignment based on any word segmentation. Therefore, our approach can be naturally scaled to syntax-based system by converting character alignment to word alignment where the word segmentation is consistent with the parsers.

We compare *CA* with the conventional word alignment (*WA*) as follows: We hand-align some sentence pairs as the evaluation set based on characters (*ESChar*), and converted it to the evaluation set based on word (*ESWord*) using the above conversion method. It is worth noting that comparing *CA* and *WA* by evaluating *CA* on *ESChar* and evaluating *WA* on *ESWord* is meaningless, because the basic tokens in *CA* and *WA* are different. However, based on the conversion method, comparing *CA* with *WA* can be accomplished by evaluating both *CA'* and *WA* on *ESWord*.

4 Experiments

4.1 Setup

FBIS corpus (LDC2003E14) (210K sentence pairs) was used for small-scale task. A large bilingual corpus of our lab (1.9M sentence pairs) was used for large-scale task. The NIST'06 and NIST'08 test sets were used as the development set and test set respectively. The Chinese portions of all these data were preprocessed by character segmenter (CHAR), ICTCLAS word segmenter[5] (ICT) and Stanford word segmenters with CTB and PKU specifications[6] respectively. The first 100 sentence pairs of the hand-aligned set in Haghighi et al. (2009) were hand-aligned as *ESChar*, which is converted to three *ESWord*s based on three segmentations respectively. These *ESWord*s were appended to training corpus with the corresponding word segmentation for evaluation purpose.

Both character and word alignment were performed by GIZA++ (Och and Ney, 2003) enhanced with *gdf* heuristics to combine bidirectional alignments (Koehn et al., 2003). A 5-gram language model was trained from the Xinhua portion of Gigaword corpus. A phrase-based MT decoder similar to (Koehn et al., 2007) was used with the decoding weights optimized by MERT (Och, 2003).

4.2 Evaluation

We first evaluate the alignment quality. The method discussed in section 3 was used to compare character and word alignment. As can be seen from Table 3, the systems using character as WSA outperformed the ones using word as WSA in both small-scale (row 3-5) and large-scale task (row 6-8) with all segmentations. This gain can be attributed to the small vocabulary size (sparsity) for character alignment. The observation is consistent with Koehn (2005) which claimed that there is a negative correlation between the vocabulary size and translation performance without explicitly distinguishing WSA and WSR.

We then evaluated the translation performance. The baselines are fully word-based MT systems (*WordSys*), i.e. using word as both WSA and WSR, and fully character-based systems (*CharSys*). Table

[4] They used this conversion for word alignment combination only, no translation results were reported.

[5] http://www.ictclas.org/

[6] http://nlp.stanford.edu/software/segmenter.shtml

		Word alignment			Character alignment		
		P	**R**	**F**	**P**	**R**	**F**
S	**CTB**	76.0	81.9	78.9	78.2	85.2	81.8
	PKU	76.1	82.0	79.0	78.0	86.1	81.9
	ICT	75.2	80.8	78.0	78.7	86.3	82.3
L	**CTB**	79.6	85.6	82.5	82.2	90.6	86.2
	PKU	80.0	85.4	82.6	81.3	89.5	85.2
	ICT	80.0	85.0	82.4	81.3	89.7	85.3

Table 3 Alignment evaluation. Precision (P), recall (R), and *F-score* (F) with $\alpha = 0.5$ (Fraser and Marcu, 2007)

	WSA	**WSR**	**CTB**	**PKU**	**ICT**
S	word	word	21.52	20.99	20.95
	char	word	**22.04**	**21.98**	**22.04**
L	word	word	22.07	22.86	22.23
	char	word	**23.41**	**23.51**	**23.05**

Table 4 Translation evaluation of *WordSys* and proposed system using BLEU-SBP (Chiang et al., 2008)

4 compares *WordSys* to our proposed system. Significant testing was carried out using bootstrap re-sampling method proposed by Koehn (2004) with a 95% confidence level. We see that our proposed systems outperformed *WordSys* in all segmentation specifications settings. Table 5 lists the results of *CharSys* in small-scale task. In this setting, we gradually set the phrase length and the distortion limits of the phrase-based decoder (context size) to 7, 9, 11 and 13, in order to remove the disadvantage of shorter context size of using character as WSR for fair comparison with *WordSys* as suggested by Duan et al. (2010). Comparing Table 4 and 5, we see that all *CharSys* underperformed *WordSys*. This observation is consistent with Chang et al. (2008) which claimed that using characters, even with large phrase length (up to 13 in our experiment) cannot always capture everything a Chinese word segmenter can do, and using word for translation is quite necessary. We also see that *CharSys* underperformed our proposed systems, that's because the harm of using character as WSR outweighed the benefit of using character as WSA, which indicated that word segmentation better for alignment is not necessarily better for translation, and vice versa.

We finally compared our approaches to Ma et al. (2007) and Ma and Way (2009), which proposed "packed word (PW)" and "bilingual motivated word (BS)" respectively. Both methods iteratively learn word segmentation and alignment alternatively, with the former starting from word-based corpus and the latter starting from characters-based corpus. Therefore, PW can be experimented on all segmentations. Table 6 lists their results in small-scale task, we see that both PW and BS underperformed our approach. This may be attributed to the low recall of the learned BS or PW in their approaches. BS underperformed both two baselines, one reason is that Ma and Way (2009) also employed word lattice decoding techniques (Dyer et al., 2008) to tackle the low recall of BS, which was removed from our experiments for fair comparison.

Context Size	**7**	**9**	**11**	**13**
BLEU	20.90	21.19	20.89	21.09

Table 5 Translation evaluation of *CharSys*.

System	**WSA**	**WSR**	**CTB**	**PKU**	**ICT**
WordSys	word	word	21.52	20.99	20.95
Proposed	char	word	**22.04**	**21.98**	**22.04**
PW	PW	PW	21.24	21.24	21.19
Char+PW	char	PW	**22.46**	**21.87**	**21.97**
BS	BS	BS		19.76	
Char+BS	char	BS		20.19	

Table 6 Comparison with other works

Interestingly, we found that using character as WSA and BS as WSR (Char+BS), a moderate gain (+0.43 point) was achieved compared with fully BS-based system; and using character as WSA and PW as WSR (Char+PW), significant gains were achieved compared with fully PW-based system, the result of CTB segmentation in this setting even outperformed our proposed approach (+0.42 point). This observation indicated that in our framework, better combinations of WSA and WSR can be found to achieve better translation performance.

5 Conclusions and Future Work

We proposed a SMT framework that uses character for alignment and word for translation, which improved both alignment quality and translation performance. We believe that in this framework, using other finer-grained segmentation, with fewer ambiguities than character, would better parameterize the alignment models, while using other coarser-grained segmentation as WSR can help capture more linguistic knowledge than word to get better translation. We also believe that our approach, if integrated with combination techniques (Dyer et al., 2008; Xi et al., 2011), can yield better results.

Acknowledgments

We thank ACL reviewers. This work is supported by the National Natural Science Foundation of China (No. 61003112), the National Fundamental Research Program of China (2010CB327903).

References

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

Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of third workshop on SMT*, pages 224-232.

David Chiang, Steve DeNeefe, Yee Seng Chan and Hwee Tou Ng. 2008. Decomposability of Translation Metrics for Improved Evaluation and Efficient Algorithms. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 610-619.

Tagyoung Chung and Daniel Gildea. 2009. Unsupervised tokenization for machine translation. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 718-726.

Michael Collins. 1999. Head-driven statistical models for natural language parsing. *Ph.D. thesis*, *University of Pennsylvania*.

Xiangyu Duan, Min Zhang, and Haizhou Li. 2010. Pseudo-word for phrase-based machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 148-156.

Christopher Dyer, Smaranda Muresan, and Philip Resnik. 2008. Generalizing word lattice translation. In *Proceedings of the Association for Computational Linguistics*, pages 1012-1020.

Jakob Elming and Nizar Habash. 2007. Combination of statistical word alignments based on multiple preprocessing schemes. In *Proceedings of the Association for Computational Linguistics*, pages 25-28.

Alexander Fraser and Daniel Marcu. 2007. Squibs and Discussions: Measuring Word Alignment Quality for Statistical Machine Translation. In *Computational Linguistics*, 33(3), pages 293-303.

Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised ITG models. In *Proceedings of the Association for Computational Linguistics*, pages 923-931.

Phillip 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, E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 177-180.

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.

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

Adam David Lopez. 2008. Machine translation by pattern matching. *Ph.D. thesis, University of Maryland.*

Yanjun Ma, Nicolas Stroppa, and Andy Way. 2007. Bootstrapping word alignment via word packing. In *Proceedings of the Association for Computational Linguistics*, pages 304-311.

Yanjun Ma and Andy Way. 2009. Bilingually motivated domain-adapted word segmentation for statistical machine translation. In *Proceedings of the Conference of the European Chapter of the ACL*, pages 549-557.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Association for Computational Linguistics*, pages 440-447.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1), pages 19-51.

Michael Paul, Andrew Finch and Eiichiro Sumita. 2010. Integration of multiple bilingually-learned segmentation schemes into statistical machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 400-408.

Jörg Tiedemann. 2009. Character-based PSMT for closely related languages. In *Proceedings of the Annual Conference of the European Association for machine Translation*, pages 12-19.

David Vilar, Jan-T. Peter and Hermann Ney. 2007. Can we translate letters? In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 33-39.

Xinyan Xiao, Yang Liu, Young-Sook Hwang, Qun Liu and Shouxun Lin. 2010. Joint tokenization and translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1200-1208.

Ning Xi, Guangchao Tang, Boyuan Li, and Yinggong Zhao. 2011. Word alignment combination over multiple word segmentation. In *Proceedings of the ACL 2011 Student Session*, pages 1-5.

Ruiqiang Zhang, Keiji Yasuda, and Eiichiro Sumita. 2008. Improved statistical machine translation by multiple Chinese word segmentation. In *Proceedings*

of the Third Workshop on Statistical Machine Translation, pages 216-223.

Translation Model Size Reduction for Hierarchical Phrase-based Statistical Machine Translation

Seung-Wook Lee[†] **Dongdong Zhang**[‡] **Mu Li**[‡] **Ming Zhou**[‡] **Hae-Chang Rim**[†]
[†] Dept. of Computer & Radio Comms. Engineering, Korea University, Seoul, South Korea
{swlee,rim}@nlp.korea.ac.kr
[‡] Microsoft Research Asia, Beijing, China
{dozhang,muli,mingzhou}@microsoft.com

Abstract

In this paper, we propose a novel method of reducing the size of translation model for hierarchical phrase-based machine translation systems. Previous approaches try to prune infrequent entries or unreliable entries based on statistics, but cause a problem of reducing the translation coverage. On the contrary, the proposed method try to prune only ineffective entries based on the estimation of the information redundancy encoded in phrase pairs and hierarchical rules, and thus preserve the search space of SMT decoders as much as possible. Experimental results on Chinese-to-English machine translation tasks show that our method is able to reduce almost the half size of the translation model with very tiny degradation of translation performance.

1 Introduction

Statistical Machine Translation (SMT) has gained considerable attention during last decades. From a bilingual corpus, all translation knowledge can be acquired automatically in SMT framework. Phrase-based model (Koehn et al., 2003) and hierarchical phrase-based model (Chiang, 2005; Chiang, 2007) show state-of-the-art performance in various language pairs. This achievement is mainly benefit from huge size of translational knowledge extracted from sufficient parallel corpus. However, the errors of automatic word alignment and non-parallelized bilingual sentence pairs sometimes have caused the unreliable and unnecessary translation rule acquisition. According to Bloodgood and Callison-Burch (2010) and our own preliminary experiments, the size of phrase table and hierarchical rule table consistently increases linearly with the growth of training size, while the translation performance tends to gain minor improvement after a certain point. Consequently, the model size reduction is necessary and meaningful for SMT systems if it can be performed without significant performance degradation. The smaller the model size is, the faster the SMT decoding speed is, because there are fewer hypotheses to be investigated during decoding. Especially, in a limited environment, such as mobile device, and for a time-urgent task, such as speech-to-speech translation, the compact size of translation rules is required. In this case, the model reduction would be the one of the main techniques we have to consider.

Previous methods of reducing the size of SMT model try to identify infrequent entries (Zollmann et al., 2008; Huang and Xiang, 2010). Several statistical significance testing methods are also examined to detect unreliable noisy entries (Tomeh et al., 2009; Johnson et al., 2007; Yang and Zheng, 2009). These methods could harm the translation performance due to their side effect of algorithms; similar multiple entries can be pruned at the same time deteriorating potential coverage of translation. The proposed method, on the other hand, tries to measure the redundancy of phrase pairs and hierarchical rules. In this work, redundancy of an entry is defined as its translational ineffectiveness, and estimated by comparing scores of entries and scores of their substituents. Suppose that the source phrase $s_1 s_2$ is always translated into $t_1 t_2$ with phrase entry $<s_1 s_2 \rightarrow t_1 t_2>$ where s_i and t_i are correspond-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 291–295,
Jeju, Republic of Korea, 8-14 July 2012.

ing translations. Similarly, source phrases s_1 and s_2 are always translated into t_1 and t_2, with phrase entries, $<s_1 \rightarrow t_1>$ and $<s_2 \rightarrow t_2>$, respectively. In this case, it is intuitive that $<s_1 s_2 \rightarrow t_1 t_2>$ could be unnecessary and redundant since its substituent always produces the same result. This paper presents statistical analysis of this redundancy measurement. The redundancy-based reduction can be performed to prune the phrase table, the hierarchical rule table, and both. Since the similar translation knowledge is accumulated at both of tables during the training stage, our reduction method performs effectively and safely. Unlike previous studies solely focus on either phrase table or hierarchical rule table, this work is the first attempt to reduce phrases and hierarchical rules simultaneously.

2 Proposed Model

Given an original translation model, TM, our goal is to find the optimally reduced translation model, TM^*, which minimizes the degradation of translation performance. To measure the performance degradation, we introduce a new metric named *consistency*:

$$C(TM, TM^*) = BLEU(D(s; TM), D(s; TM^*)) \quad (1)$$

where the function D produces the target sentence of the source sentence s, given the translation model TM. *Consistency* measures the similarity between the two groups of decoded target sentences produced by two different translation models. There are number of similarity metrics such as Dices coefficient (Kondrak et al., 2003), and Jaccard similarity coefficient. Instead, we use BLEU scores (Papineni et al., 2002) since it is one of the primary metrics for machine translation evaluation. Note that our *consistency* does not require the reference set while the original BLEU does. This means that only (abundant) source-side monolingual corpus is needed to predict performance degradation. Now, our goal can be rewritten with this metric; among all the possible reduced models, we want to find the set which can maximize the *consistency*:

$$TM^* = \underset{TM' \subset TM}{argmax}\ C(TM, TM') \quad (2)$$

In minimum error rate training (MERT) stages, a development set, which consists of bilingual sentences, is used to find out the best weights of features (Och, 2003). One characteristic of our method is that it isolates feature weights of the translation model from SMT log-linear model, trying to minimize the impact of search path during decoding. The reduction procedure consists of three stages: translation scoring, redundancy estimation, and redundancy-based reduction.

Our reduction method starts with measuring the translation scores of the individual phrase and the hierarchical rule. Similar to the decoder, the scoring scheme is based on the log-linear framework:

$$PS(p) = \sum_i \lambda_i h_i(p) \quad (3)$$

where h is a feature function and λ is its weight. As the conventional hierarchical phrase-based SMT model, our features are composed of $P(e|f)$, $P(f|e)$, $P_{lex}(e|f)$, $P_{lex}(f|e)$, and the number of phrases, where e and f denote a source phrase and a target phrase, respectively. P_{lex} is the lexicalized probability. In a similar manner, the translation scores of hierarchical rules are calculated as follows:

$$HS(r) = \sum_i \lambda_i h_i(r) \quad (4)$$

The features are as same as those that are used for phrase scoring, except the last feature. Instead of the phrase number penalty, the hierarchical rule number penalty is used. The weight for each feature is shared from the results of MERT. With this scoring scheme, our model is able to measure how important the individual entry is during decoding.

Once translation scores for all entries are estimated, our method retrieves substituent candidates with their combination scores. The combination score is calculated by accumulating translation scores of every member as follows:

$$CS(p_{1...n}) = \sum_{i=1}^{n} PS(p_i) \quad (5)$$

This scoring scheme follows the same manner what the conventional decoder does, finding the best phrase combination during translation. By comparing the original translation score with combination

scores of its substituents, the redundancy scores are estimated, as follows:

$$Red(p) = \min_{p_{1...n} \in Sub(p)} PS(p) - CS(p_{1...n}) \quad (6)$$

where Sub is the function that retrieves all possible substituents (the combinations of sub-phrases, and/or sub-rules that exactly produce the same target phrase, given the source phrase p). If the combination score of the best substituent is same as the translation score of p, the redundancy score becomes zero. In this case, the decoder always produces the same translation results without p. When the redundancy score is negative, the best substituent is more likely to be chosen instead of p. This implies that there is no risk to prune p; the search space is not changed, and the search path is not changed as well.

Our method can be varied according to the designation of Sub function. If both of the phrase table and the hierarchical rule table are allowed, cross reduction can be possible; the phrase table is reduced based on the hierarchical rule table and vice versa. With extensions of combination scoring and redundancy scoring schemes like following equations, our model is able to perform cross reduction.

$$CS(p_{1...n}, h_{1...m}) = \sum_{i=1}^{n} PS(p_i) + \sum_{i=1}^{m} HS(h_i) \quad (7)$$

$$Red(p) = \min_{<p_{1...n}, h_{1...m}> \in Sub(p)} PS(p) - CS(p_{1...n}, h_{1...m}) \quad (8)$$

The proposed method has some restrictions for reduction. First of all, it does not try to prune the phrase that has no substituents, such as unigram phrases; the phrase whose source part is composed of a single word. This restriction guarantees that the translational coverage of the reduced model is as high as those of the original translation model. In addition, our model does not prune the phrases and the hierarchical rules that have reordering within it to prevent information loss of reordering. For instance, if we prune phrase, $<s_1 s_2 s_3 \rightarrow t_3 t_1 t_2>$, phrases, $<s_1 s_2 \rightarrow t_1 t_2>$ and $<s_3 \rightarrow t_3>$ are not able to produce the same target words without appropriate reordering.

Once the redundancy scores for all entries have been estimated, the next step is to select the best N entries to prune to satisfy a desired model size. We can simply prune the first N from the list of entries sorted by increasing order of redundancy score. However, this method may not result in the optimal reduction, since each redundancy scores are estimated based on the assumption of the existence of all the other entries. In other words, there are dependency relationships among entries. We examine two methods to deal with this problem. The first is to ignore dependency, which is the more efficient manner. The other is to prune independent entries first. After all independent entries are pruned, the dependent entries are started to be pruned. We present the effectiveness of each method in the next section.

Since our goal is to reduce the size of all translation models, the reduction is needed to be performed for both the phrase table and the hierarchical rule table simultaneously, namely joint reduction. Similar to phrase reduction and hierarchical rule reduction, it selects the best N entries of the mixture of phrase and hierarchical rules. This method results in safer pruning; once a phrase is determined to be pruned, the hierarchical rules, which are related to this phrase, are likely to be kept, and vice versa.

3 Experiment

We investigate the effectiveness of our reduction method by conducting Chinese-to-English translation task. The training data, as same as Cui et al. (2010), consists of about 500K parallel sentence pairs which is a mixture of several datasets published by LDC. NIST 2003 set is used as a development set. NIST 2004, 2005, 2006, and 2008 sets are used for evaluation purpose. For word alignment, we use GIZA++[1], an implementation of IBM models (Brown et al., 1993). We have implemented a hierarchical phrase-based SMT model similar to Chiang (2005). The trigram target language model is trained from the Xinhua portion of English Gigaword corpus (Graff and Cieri, 2003). Sampled 10,000 sentences from Chinese Gigaword corpus (Graff, 2007) was used for source-side development dataset to measure consistency. Our main metric for translation performance evaluation is case-

[1] http://www.statmt.org/moses/giza/GIZA++.html

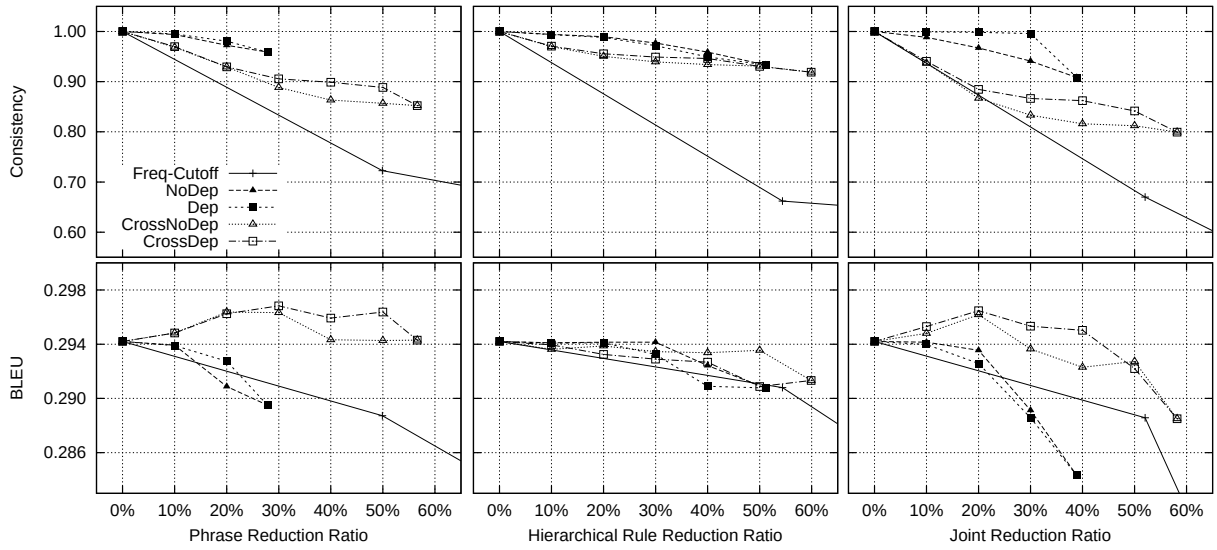


Figure 1: Performance comparison. BLEU scores and consistency scores are averaged over four evaluation sets.

insensitive BLEU-4 scores (Papineni et al., 2002).

As a baseline system, we chose the frequency-based cutoff method, which is one of the most widely used filtering methods. As shown in Figure 1, almost half of the phrases and hierarchical rules are pruned when cutoff=2, while the BLEU score is also deteriorated significantly. We introduced two methods for selecting the N pruning entries considering dependency relationships. The non-dependency method does not consider dependency relationships, while the dependency method prunes independent entries first. Each method can be combined with cross reduction. The performance is measured in three different reduction tasks: phrase reduction, hierarchical rule reduction, and joint reduction. As the reduction ratio becomes higher, the model size, i.e., the number of entries, is reduced while BLEU scores and coverage are decreased. The results show that the translation performance is highly co-related with the *consistency*. The co-relation scores measured between them on the phrase reduction and the hierarchical rule reduction tasks are 0.99 and 0.95, respectively, which indicates very strong positive relationship.

For the phrase reduction task, the dependency method outperforms the non-dependency method in terms of BLEU score. When the cross reduction technique was used for the phrase reduction task, BLEU score is not deteriorated even when more than half of phrase entries are pruned. This result implies that there is much redundant information stored in the hierarchical rule table. On the other hand, for the hierarchical rule reduction task, the non-dependency method shows the better performance. The dependency method sometimes performs worse than the baseline method. We expect that this is caused by the unreliable estimation of dependency among hierarchical rules since the most of them are automatically generated from the phrases. The excessive dependency of these rules would cause overestimation of hierarchical rule redundancy score.

4 Conclusion

We present a novel method of reducing the size of translation model for SMT. The contributions of the proposed method are as follows: 1) our method is the first attempt to reduce the phrase table and the hierarchical rule table simultaneously. 2) our method is a safe reduction method since it considers the redundancy, which is the practical ineffectiveness of individual entry. 3) our method shows that almost the half size of the translation model can be reduced without significant performance degradation. It may be appropriate for the applications running on limited environment, e.g., mobile devices.

Acknowledgement

The first author performed this research during an internship at Microsoft Research Asia. This research was supported by the MKE(The Ministry of Knowledge Economy), Korea and Microsoft Research, under IT/SW Creative research program supervised by the NIPA(National IT Industry Promotion Agency). (NIPA-2010-C1810-1002-0025)

References

Michael Bloodgood and Chris Callison-Burch. 2010. Bucking the Trend: Large-Scale Cost-Focused Active Learning for Statistical Machine Translation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 854–864.

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*, 19:263–311, June.

David Chiang. 2005. A Hierarchical Phrase-based Model for Statistical Machine Translation. In *Proceedings of the 43th Annual Meeting on Association for Computational Linguistics*, pages 263–270.

David Chiang. 2007. Hierarchical Phrase-based Translation. *Computational Linguistics*, 33:201–228, June.

Lei Cui, Dongdong Zhang, Mu Li, Ming Zhou, and Tiejun Zhao. 2010. Hybrid Decoding: Decoding with Partial Hypotheses Combination Over Multiple SMT Systems. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 214–222, Stroudsburg, PA, USA. Association for Computational Linguistics.

David Graff and Christopher Cieri. 2003. English Gigaword. In *Linguistic Data Consortium, Philadelphia*.

David Graff. 2007. Chinese Gigaword Third Edition. In *Linguistic Data Consortium, Philadelphia*.

Fei Huang and Bing Xiang. 2010. Feature-Rich Discriminative Phrase Rescoring for SMT. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 492–500, Stroudsburg, PA, USA. Association for Computational Linguistics.

Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. 2007. Improving Translation Quality by Discarding Most of the Phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 967–975.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical Phrase-based Translation. 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 48–54, Stroudsburg, PA, USA. Association for Computational Linguistics.

Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can Improve Statistical Translation Models. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003–short papers - Volume 2*, NAACL-Short '03, pages 46–48, Stroudsburg, PA, USA. 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 - Volume 1*, ACL '03, pages 160–167, Stroudsburg, PA, USA. 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 the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Nadi Tomeh, Nicola Cancedda, and Marc Dymetman. 2009. Complexity-based Phrase-Table Filtering for Statistical Machine Translation.

Mei Yang and Jing Zheng. 2009. Toward Smaller, Faster, and Better Hierarchical Phrase-based SMT. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 237–240, Stroudsburg, PA, USA. 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, troudsburg, PA, USA. Association for Computational Linguistics.

Heuristic Cube Pruning in Linear Time

Andrea Gesmundo
Department of
Computer Science
University of Geneva
andrea.gesmundo@unige.ch

Giorgio Satta
Department of
Information Engineering
University of Padua
satta@dei.unipd.it

James Henderson
Department of
Computer Science
University of Geneva
james.henderson@unige.ch

Abstract

We propose a novel heuristic algorithm for Cube Pruning running in linear time in the beam size. Empirically, we show a gain in running time of a standard machine translation system, at a small loss in accuracy.

1 Introduction

Since its first appearance in (Huang and Chiang, 2005), the Cube Pruning (CP) algorithm has quickly gained popularity in statistical natural language processing. Informally, this algorithm applies to scenarios in which we have the k-best solutions for two input sub-problems, and we need to compute the k-best solutions for the new problem representing the combination of the two sub-problems.

CP has applications in tree and phrase based machine translation (Chiang, 2007; Huang and Chiang, 2007; Pust and Knight, 2009), parsing (Huang and Chiang, 2005), sentence alignment (Riesa and Marcu, 2010), and in general in all systems combining inexact beam decoding with dynamic programming under certain monotonic conditions on the definition of the scores in the search space.

Standard implementations of CP run in time $\mathcal{O}(k \log(k))$, with k being the size of the input/output beams (Huang and Chiang, 2005). Gesmundo and Henderson (2010) propose Faster CP (FCP) which optimizes the algorithm but keeps the $\mathcal{O}(k \log(k))$ time complexity. Here, we propose a novel heuristic algorithm for CP running in time $\mathcal{O}(k)$ and evaluate its impact on the efficiency and performance of a real-world machine translation system.

2 Preliminaries

Let $\mathcal{L} = \langle x_0, \ldots, x_{k-1} \rangle$ be a list over $\mathbb{R}$, that is, an ordered sequence of real numbers, possibly with repetitions. We write $|\mathcal{L}| = k$ to denote the length of $\mathcal{L}$. We say that $\mathcal{L}$ is **descending** if $x_i \geq x_j$ for every i, j with $0 \leq i < j < k$. Let $\mathcal{L}_1 = \langle x_0, \ldots, x_{k-1} \rangle$ and $\mathcal{L}_2 = \langle y_0, \ldots, y_{k'-1} \rangle$ be two descending lists over $\mathbb{R}$. We write $\mathcal{L}_1 \oplus \mathcal{L}_2$ to denote the descending list with elements $x_i + y_j$ for every i, j with $0 \leq i < k$ and $0 \leq j < k'$.

In **cube pruning** (CP) we are given as input two descending lists $\mathcal{L}_1$, $\mathcal{L}_2$ over $\mathbb{R}$ with $|\mathcal{L}_1| = |\mathcal{L}_2| = k$, and we are asked to compute the descending list consisting of the first k elements of $\mathcal{L}_1 \oplus \mathcal{L}_2$.

A problem related to CP is the **k-way merge** problem (Horowitz and Sahni, 1983). Given descending lists $\mathcal{L}_i$ for every i with $0 \leq i < k$, we write $\mathsf{merge}_{i=0}^{k-1}\, \mathcal{L}_i$ to denote the "merge" of all the lists $\mathcal{L}_i$, that is, the descending list with all elements from the lists $\mathcal{L}_i$, including repetitions.

For $\Delta \in \mathbb{R}$ we define $\mathsf{shift}(\mathcal{L}, \Delta) = \mathcal{L} \oplus \langle \Delta \rangle$. In words, $\mathsf{shift}(\mathcal{L}, \Delta)$ is the descending list whose elements are obtained by "shifting" the elements of $\mathcal{L}$ by Δ, preserving the order. Let $\mathcal{L}_1, \mathcal{L}_2$ be descending lists of length k, with $\mathcal{L}_2 = \langle y_0, \ldots, y_{k-1} \rangle$. Then we can express the output of CP on $\mathcal{L}_1, \mathcal{L}_2$ as the list

$$\mathsf{merge}_{i=0}^{k-1}\, \mathsf{shift}(\mathcal{L}_1, y_i) \qquad (1)$$

truncated after the first k elements. This shows that the CP problem is a particular instance of the k-way merge problem, in which all input lists are related by k *independent* shifts.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 296–300,
Jeju, Republic of Korea, 8-14 July 2012.

Computation of the solution of the k-way merge problem takes time $\mathcal{O}(q \log(k))$, where q is the size of the output list. In case each input list has length k this becomes $\mathcal{O}(k^2 \log(k))$, and by restricting the computation to the first k elements, as required by the CP problem, we can further reduce to $\mathcal{O}(k \log(k))$. This is the already known upper bound on the CP problem (Huang and Chiang, 2005; Gesmundo and Henderson, 2010). Unfortunately, there seems to be no way to achieve an asymptotically faster algorithm by exploiting the restriction that the input lists are all related by some shifts. Nonetheless, in the next sections we use the above ideas to develop a heuristic algorithm running in time linear in k.

3 Cube Pruning With Constant Slope

Consider lists $\mathcal{L}_1, \mathcal{L}_2$ defined as in section 2. We say that $\mathcal{L}_2$ has **constant slope** if $y_{i-1} - y_i = \Delta > 0$ for every i with $0 < i < k$. Throughout this section we assume that $\mathcal{L}_2$ has constant slope, and we develop an (exact) linear time algorithm for solving the CP problem under this assumption.

For each $i \geq 0$, let I_i be the left-open interval $(x_0 - (i+1) \cdot \Delta,\ x_0 - i \cdot \Delta]$ of $\mathbb{R}$. Let also $s = \lfloor (x_0 - x_{k-1})/\Delta \rfloor + 1$. We split $\mathcal{L}_1$ into (possibly empty) sublists σ_i, $0 \leq i < s$, called **segments**, such that each σ_i is the descending sublist consisting of all elements from $\mathcal{L}_1$ that belong to I_i. Thus, moving down one segment in $\mathcal{L}_1$ is the closest equivalent to moving down one element in $\mathcal{L}_2$.

Let $t = \min\{k, s\}$; we define descending lists M_i, $0 \leq i < t$, as follows. We set $M_0 = \mathsf{shift}(\sigma_0, y_0)$, and for $1 \leq i < t$ we let

$$M_i = \mathsf{merge}\{\mathsf{shift}(\sigma_i, y_0), \mathsf{shift}(M_{i-1}, -\Delta)\} \quad (2)$$

We claim that the ordered concatenation of M_0, $M_1, \ldots, M_{t-1}$ truncated after the first k elements is exactly the output of CP on input $\mathcal{L}_1, \mathcal{L}_2$.

To prove our claim, it helps to visualize the descending list $\mathcal{L}_1 \oplus \mathcal{L}_2$ (of size k^2) as a $k \times k$ matrix L whose j-th column is $\mathsf{shift}(\mathcal{L}_1, y_j)$, $0 \leq j < k$. For an interval $I = (x, x']$, we define $\mathsf{shift}(I, y) = (x + y, x' + y]$. Similarly to what we have done with $\mathcal{L}_1$, we can split each column of L into s segments. For each i, j with $0 \leq i < s$ and $0 \leq j < k$, we define the i-th segment of the j-th column, written $\sigma_{i,j}$, as the descending sublist consisting of all elements of that column that belong to $\mathsf{shift}(I_i, y_j)$. Then we have $\sigma_{i,j} = \mathsf{shift}(\sigma_i, y_j)$.

For any d with $0 \leq d < t$, consider now all segments $\sigma_{i,j}$ with $i + j = d$, forming a sub-antidiagonal in L. We observe that these segments contain *all and only* those elements of L that belong to the interval I_d. It is not difficult to show by induction that these elements are exactly the elements that appear in descending order in the list M_i defined in (2).

We can then directly use relation (2) to iteratively compute CP on two lists of length k, under our assumption that one of the two lists has constant slope. Using the fact that the merge of two lists as in (2) can be computed in time linear in the size of the output list, it is not difficult to implement the above algorithm to run in time $\mathcal{O}(k)$.

4 Linear Time Heuristic Solution

In this section we further elaborate on the exact algorithm of section 3 for the constant slope case, and develop a heuristic solution for the general CP problem. Let $\mathcal{L}_1, \mathcal{L}_2, L$ and k be defined as in sections 2 and 3. Despite the fact that $\mathcal{L}_2$ does not have a constant slope, we can still split each column of L into segments, as follows.

Let $\widetilde{I}_i$, $0 \leq i < k-1$, be the left-open interval $(x_0 + y_{i+1},\ x_0 + y_i]$ of $\mathbb{R}$. Note that, unlike the case of section 3, intervals $\widetilde{I}_i$'s are not all of the same size now. Let also $\widetilde{I}_{k-1} = [x_{k-1} + y_{k-1},\ x_0 + y_{k-1}]$. For each i, j with $0 \leq j < k$ and $0 \leq i < k - j$, we define segment $\widetilde{\sigma}_{i,j}$ as the descending sublist consisting of all elements of the j-th column of L that belong to $\widetilde{I}_{i+j}$. In this way, the j-th column of L is split into segments $\widetilde{I}_j, \widetilde{I}_{j+1}, \ldots, \widetilde{I}_{k-1}$, and we have a variable number of segments per column. Note that segments $\widetilde{\sigma}_{i,j}$ with a constant value of $i+j$ contain *all and only* those elements of L that belong to the left-open interval $\widetilde{I}_{i+j}$.

Similarly to section 3, we define descending lists $\widetilde{M}_i$, $0 \leq i < k$, by setting $\widetilde{M}_0 = \widetilde{\sigma}_{0,0}$ and, for $1 \leq i < k$, by letting

$$\widetilde{M}_i = \mathsf{merge}\{\widetilde{\sigma}_{i,0}\ ,\ \mathsf{path}(\widetilde{M}_{i-1}, L)\} \quad (3)$$

Note that the function $\mathsf{path}(\widetilde{M}_{i-1}, L)$ should not return $\mathsf{shift}(\widetilde{M}_{i-1}, -\Delta)$, for some value Δ, as in the

1: **Algorithm 1** $(\mathcal{L}_1, \mathcal{L}_2) : \widetilde{\mathcal{L}}^\star$
2: $\widetilde{\mathcal{L}}^\star$.insert($L[0,0]$);
3: referColumn $\leftarrow$ 0;
4: $x_{follow} \leftarrow L[0,1]$;
5: $x_{deviate} \leftarrow L[1,0]$;
6: $\mathcal{C} \leftarrow$ CircularList($[0,1]$);
7: $\mathcal{C}$-iterator $\leftarrow \mathcal{C}$.begin();
8: **while** $|\widetilde{\mathcal{L}}^\star| < k$ **do**
9: **if** $x_{follow} > x_{deviate}$ **then**
10: $\widetilde{\mathcal{L}}^\star$.insert($x_{follow}$);
11: **if** $\mathcal{C}$-iterator.current()=$[0,1]$ **then**
12: referColumn++;
13: $[i,j] \leftarrow \mathcal{C}$-iterator.next();
14: $x_{follow} \leftarrow L[i$,referColumn$+j]$;
15: **else**
16: $\widetilde{\mathcal{L}}^\star$.insert($x_{deviate}$);
17: $i \leftarrow x_{deviate}$.row();
18: $\mathcal{C}$-iterator.insert($[i, -$referColumn$]$);
19: $x_{deviate} \leftarrow L[i+1,0]$;

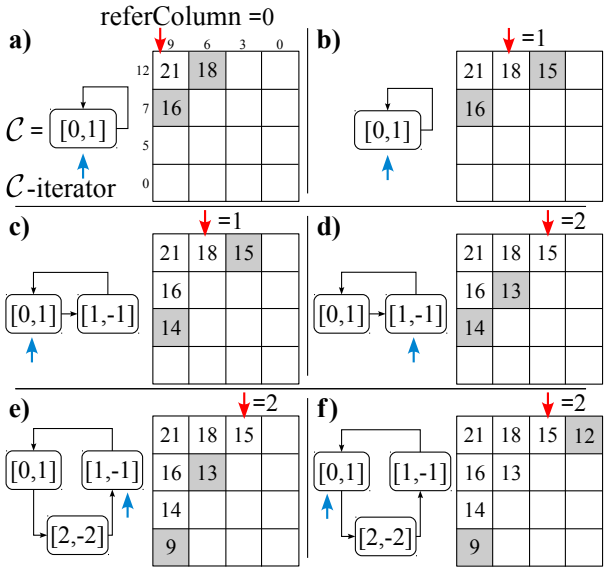


Figure 1: A running example for Algorithm 1.

case of (2). This is because input list $\mathcal{L}_2$ does not have constant slope in general. In an exact algorithm, $\mathsf{path}(\widetilde{M}_{i-1}, L)$ should return the descending list $\mathcal{L}^\star_{i-1} = \mathsf{merge}_{j=1}^{i}\, \widetilde{\sigma}_{i-j,j}$: Unfortunately, we do not know how to compute such a i-way merge without introducing a logarithmic factor.

Our solution is to define $\mathsf{path}(\widetilde{M}_{i-1}, L)$ in such a way that it computes a list $\widetilde{\mathcal{L}}_{i-1}$ which is a permutation of the correct solution $\mathcal{L}^\star_{i-1}$. To do this, we consider the "relative" path starting at $x_0 + y_{i-1}$ that we need to follow in L in order to collect all the elements of $\widetilde{M}_{i-1}$ in the given order. We then apply such a path starting at $x_0 + y_i$ and return the list of collected elements. Finally, we compute the output list $\widetilde{\mathcal{L}}^\star$ as the concatenation of all lists $\widetilde{M}_i$ up to the first k elements.

It is not difficult to see that when $\mathcal{L}_2$ has constant slope we have $\widetilde{M}_i = M_i$ for all i with $0 \le i < k$, and list $\widetilde{\mathcal{L}}^\star$ is the exact solution to the CP problem. When $\mathcal{L}_2$ does not have a constant slope, list $\widetilde{\mathcal{L}}^\star$ might depart from the exact solution in two respects: it might not be a descending list, because of local variations in the ordering of the elements; and it might not be a permutation of the exact solution, because of local variations at the end of the list. In the next section we evaluate the impact that our heuristic solution has on the performance of a real-world machine translation system.

Algorithm 1 implements the idea presented in (3). The algorithm takes as input two descending lists $\mathcal{L}_1, \mathcal{L}_2$ of length k and outputs the list $\widetilde{\mathcal{L}}^\star$ which approximates the desired solution. Element $L[i,j]$ denotes the combined value $x_i + y_j$, and is always computed on demand.

We encode a relative path (mentioned above) as a sequence of elements, called **displacements**, each of the form $[i, \delta]$. Here i is the index of the next row, and δ represents the *relative* displacement needed to reach the next column, to be summed to a variable called referColumn denoting the index of the column of the first element of the path. The reason why only the second coordinate is a relative value is that we shift paths only horizontally (row indices are preserved). The relative path is stored in a circular list $\mathcal{C}$, with displacement $[0,1]$ marking the starting point (paths are always shifted one element to the right). When merging the list obtained through the path for $\widetilde{M}_{i-1}$ with segment $\widetilde{\sigma}_{i,0}$, as specified in (3), we update $\mathcal{C}$ accordingly, so that the new relative path can be used at the next round for $\widetilde{M}_i$. The merge operator is implemented by the while cycle at lines 8 to 19 of algorithm 1. The if statement at line 9 tests whether the next step should follow the relative path for $\widetilde{M}_{i-1}$ stored in $\mathcal{C}$ (lines 10 to 14) or

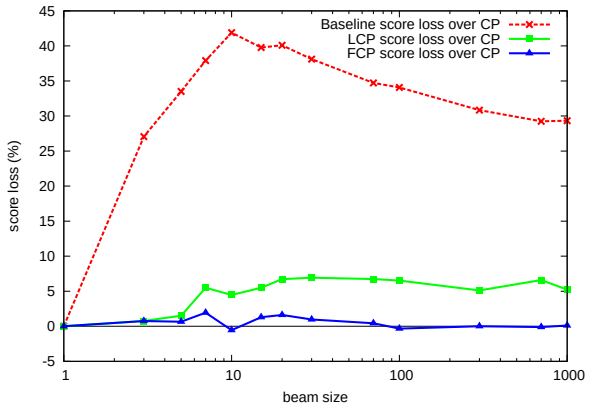


Figure 2: Search-score loss relative to standard CP.

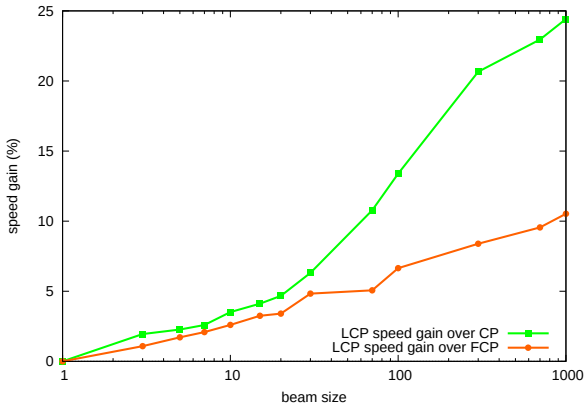


Figure 3: Linear CP relative speed gain.

else depart visiting an element from $\widetilde{\sigma}_{i,0}$ in the first column of L (lines 16 to 19). In the latter case, we update $\mathcal{C}$ with the new displacement (line 18), where the function insert() inserts a new element before the one currently pointed to. The function next() at line 13 moves the iterator to the next element and then returns its value.

A running example of algorithm 1 is reported in Figure 1. The input lists are $\mathcal{L}_1 = \langle 12, 7, 5, 0 \rangle$, $\mathcal{L}_2 = \langle 9, 6, 3, 0 \rangle$. Each of the picture in the sequence represents the state of the algorithm when the test at line 9 is executed. The value in the shaded cell in the first column is $x_{deviate}$, while the value in the other shaded cell is x_{follow}.

5 Experiments

We implement Linear CP (LCP) on top of Cdec (Dyer et al., 2010), a widely-used hierarchical MT system that includes implementations of standard CP and FCP algorithms. The experiments were executed on the NIST 2003 Chinese-English parallel corpus. The training corpus contains 239k sentence pairs. A binary translation grammar was extracted using a suffix array rule extractor (Lopez, 2007). The model was tuned using MERT (Och, 2003). The algorithms are compared on the NIST-03 test set, which contains 919 sentence pairs. The features used are basic lexical features, word penalty and a 3-gram Language Model (Heafield, 2011).

Since we compare decoding algorithms on the same search space, the accuracy comparison is done in terms of search score. For each algorithm we compute the average score of the best translation found for the test sentences. In Figure 2 we plot the score-loss relative to standard CP average score. Note that the FCP loss is always $< 3\%$, and the LCP loss is always $< 7\%$. The dotted line plots the loss of a baseline linear time heuristic algorithm which assumes that both input lists have constant slope, and that scans L along parallel lines whose steep is the ratio of the average slope of each input list. The baseline greatly deteriorates the accuracy: this shows that finding a reasonable linear time heuristic algorithm is not trivial. We can assume a bounded loss in accuracy, because for larger beam size all the algorithms tend to converge to exhaustive search.

We found that these differences in search score resulted in no significant variations in BLEU score (e.g. with $k = 30$, CP reaches 32.2 while LCP 32.3).

The speed comparison is done in terms of algorithm run-time. Figure 3 plots the relative speed gain of LCP over standard CP and over FCP. Given the log-scale used for the beam size k, the linear shape of the speed gain over FCP (and CP) in Figure 3 empirically confirms that LCP has a $\log(k)$ asymptotic advantage over FCP and CP.

In addition to Chinese-English, we ran experiments on translating English to French (from Europarl corpus (Koehn, 2005)), and find that the LCP score-loss relative to CP is $< 9\%$ while the speed relative advantage of LCP over CP increases in average by 11.4% every time the beam size is multiplied by 10 (e.g. with $k = 1000$ the speed advantage is 34.3%). These results confirm the bounded accuracy loss and $\log(k)$ speed advantage of LCP.

References

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

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Jonathan Weese, Hendra Setiawan, Ferhan Ture, Vladimir Eidelman, Phil Blunsom, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *ACL '10: Proceedings of the ACL 2010 System Demonstrations*, Uppsala, Sweden.

Andrea Gesmundo and James Henderson. 2010. Faster Cube Pruning. In *IWSLT '10: Proceedings of the 7th International Workshop on Spoken Language Translation*, Paris, France.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *WMT '11: Proceedings of the 6th Workshop on Statistical Machine Translation*, Edinburgh, Scotland, UK.

E. Horowitz and S. Sahni. 1983. *Fundamentals of data structures*. Computer software engineering series. Computer Science Press.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *IWPT '05: Proceedings of the 9th International Workshop on Parsing Technology*, Vancouver, British Columbia, Canada.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *ACL '07: Proceedings of the 45th Conference of the Association for Computational Linguistics*, Prague, Czech Republic.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th Machine Translation Summit*, Phuket, Thailand.

Adam Lopez. 2007. Hierarchical phrase-based translation with suffix arrays. In *EMNLP-CoNLL '07: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Prague, Czech Republic.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *ACL '03: Proceedings of the 41st Conference of the Association for Computational Linguistics*, Sapporo, Japan.

Michael Pust and Kevin Knight. 2009. Faster MT decoding through pervasive laziness. In *NAACL '09: Proceedings of the 10th Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, CO, USA.

Jason Riesa and Daniel Marcu. 2010. Hierarchical search for word alignment. In *ACL '10: Proceedings of the 48th Conference of the Association for Computational Linguistics*, Uppsala, Sweden.

Combining Word-Level and Character-Level Models for Machine Translation Between Closely-Related Languages

Preslav Nakov
Qatar Computing Research Institute
Qatar Foundation, P.O. box 5825
Doha, Qatar
pnakov@qf.org.qa

Jörg Tiedemann
Department of Linguistics and Philology
Uppsala University
Uppsala, Sweden
jorg.tiedemann@lingfil.uu.se

Abstract

We propose several techniques for improving statistical machine translation between closely-related languages with scarce resources. We use character-level translation trained on n-gram-character-aligned bitexts and tuned using word-level BLEU, which we further augment with character-based transliteration at the word level and combine with a word-level translation model. The evaluation on Macedonian-Bulgarian movie subtitles shows an improvement of 2.84 BLEU points over a phrase-based word-level baseline.

1 Introduction

Statistical machine translation (SMT) systems, require parallel corpora of sentences and their translations, called *bitexts*, which are often not sufficiently large. However, for many *closely-related* languages, SMT can be carried out even with small bitexts by exploring relations below the word level.

Closely-related languages such as Macedonian and Bulgarian exhibit a large overlap in their vocabulary and strong syntactic and lexical similarities. Spelling conventions in such related languages can still be different, and they may diverge more substantially at the level of morphology. However, the differences often constitute consistent regularities that can be generalized when translating.

The language similarities and the regularities in morphological variation and spelling motivate the use of character-level translation models, which were applied to translation (Vilar et al., 2007; Tiedemann, 2009a) and transliteration (Matthews, 2007).

Macedonian	Bulgarian
а в м е _	а х м е _
а в м е _ д а _	а х м е _ д а _
_ в е р у в а м _	_ в я р в а м _
_ д е к а _ т о ј _	_ , ч е _ т о й _

Table 1: Examples from a character-level phrase table (without scores): mappings can cover words and phrases.

Certainly, translation cannot be adequately modeled as simple transliteration, even for closely-related languages. However, the strength of phrase-based SMT (Koehn et al., 2003) is that it can support rather large sequences (phrases) that capture translations of entire chunks. This makes it possible to include mappings that go far beyond the edit-distance-based string operations usually modeled in transliteration. Table 1 shows how character-level phrase tables can cover mappings spanning over multi-word units. Thus, character-level phrase-based SMT models combine the generality of character-by-character transliteration and lexical mappings of larger units that could possibly refer to morphemes, words or phrases, as well as to various combinations thereof.

2 Training Character-level SMT Models

We treat sentences as sequences of characters instead of words, as shown in Figure 1. Due to the reduced vocabulary, we can use higher-order models, which is necessary in order to avoid the generation of non-word sequences. In our case, we opted for a 10-character language model and a maximum phrase length of 10 (based on initial experiments).

However, word alignment models are not fit for character-level SMT, where the vocabulary shrinks.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 301–305,
Jeju, Republic of Korea, 8-14 July 2012.

original:
MK: навистина ?
BG: наистина ?

characters:
MK: н а в и с т и н а _ ?
BG: н а и с т и н а _ ?

character bigrams:
MK: на ав ви ис ст ти ин на а_ _? ?_
BG: на аи ис ст ти ин на а_ _? ?_

Figure 1: Preparing the training corpus for alignment.

Statistical word alignment models heavily rely on context-independent lexical translation parameters and, therefore, are unable to properly distinguish character mapping differences in various contexts. The alignment models used in the transliteration literature have the same problem as they are usually based on edit distance operations and finite-state automata without contextual history (Jiampojamarn et al., 2007; Damper et al., 2005; Ristad and Yianilos, 1998). We, thus, transformed the input to sequences of character n-grams as suggested by Tiedemann (2012); examples are shown in Figure 1. This artificially increases the vocabulary as shown in Table 2, making standard alignment models and their lexical translation parameters more expressive.

	Macedonian	**Bulgarian**
single characters	99	101
character bigrams	1,851	1,893
character trigrams	13,794	14,305
words	41,816	30,927

Table 2: Vocabulary size of character-level alignment models and the corresponding word-level model.

It turns out that bigrams constitute a good compromise between generality and contextual specificity, which yields useful character alignments with good performance in terms of phrase-based translation. In our experiments, we used GIZA++ (Och and Ney, 2003) with standard settings and the *grow-diagonal-final-and* heuristics to symmetrize the final IBM-model-4-based Viterbi alignments (Brown et al., 1993). The phrases were extracted and scored using the Moses training tools (Koehn et al., 2007).[1]

We tuned the parameters of the log-linear SMT model using minimum error rate training (Och, 2003), optimizing BLEU (Papineni et al., 2002). Since BLEU over matching character sequences does not make much sense, especially if the k-gram size is limited to small values of k (usually, 4 or less), we post-processed n-best lists in each tuning step to calculate the usual word-based BLEU score.

[1]Note that the extracted phrase table does not include sequences of character n-grams. We map character n-gram alignments to links between single characters before extraction.

3 Transliteration

We also built a character-level SMT system for word-level transliteration, which we trained on a list of automatically extracted pairs of likely cognates.

3.1 Cognate Extraction

Classic NLP approaches to cognate extraction look for words with similar spelling that co-occur in parallel sentences (Kondrak et al., 2003). Since our Macedonian-Bulgarian bitext (MK–BG) was small, we further used a MK–EN and an EN–BG bitext.

First, we induced IBM-model-4 word alignments for MK–EN and EN–BG, from which we extracted four conditional lexical translation probabilities: $\Pr(m|e)$ and $\Pr(e|m)$ for MK–EN, and $\Pr(b|e)$ and $\Pr(e|b)$ for EN–BG, where m, e, and b stand for a Macedonian, an English, and a Bulgarian word.

Then, following (Callison-Burch et al., 2006; Wu and Wang, 2007; Utiyama and Isahara, 2007), we induced conditional lexical translation probabilities as $\Pr(m|b) = \sum_e \Pr(m|e)\Pr(e|b)$, where $\Pr(m|e)$ and $\Pr(e|b)$ are estimated using maximum likelihood from MK–EN and EN–BG word alignments.

Then, we induced translation probability estimations for the reverse direction $\Pr(b|m)$ and we calculated the quantity $\mathrm{Piv}(m,b) = \Pr(m|b)\Pr(b|m)$. We calculated a similar quantity $\mathrm{Dir}(m,b)$, where the probabilities $\Pr(m|b)$ and $\Pr(b|m)$ are estimated using maximum likelihood from the MK–BG bitext directly. Finally, we calculated the similarity score $S(m,b) = \mathrm{Piv}(m,b) + \mathrm{Dir}(m,b) + 2\times\mathrm{LCSR}(m,b)$, where LCSR is the longest common subsequence of two strings, divided by the length of the longer one.

The score $S(m,b)$ is high for words that are likely to be cognates, i.e., that (i) have high probability of being mutual translations, which is expressed by the first two terms in the summation, and (ii) have similar spelling, as expressed by the last term. Here we give equal weight to $\mathrm{Dir}(m,b)$ and $\mathrm{Piv}(m,b)$; we also give equal weights to the translational similarity (the sum of the first two terms) and to the spelling similarity (twice LCSR).

We excluded all words of length less than three, as well as all Macedonian-Bulgarian word pairs (m, b) for which $\mathrm{Piv}(m, b) + \mathrm{Dir}(m, b) < 0.01$, and those for which $\mathrm{LCSR}(m, b)$ was below 0.58, a value found by Kondrak et al. (2003) to work well for a number of European language pairs.

Finally, using $S(m, b)$, we induced a weighted bipartite graph, and we performed a greedy approximation to the maximum weighted bipartite matching in that graph using *competitive linking* (Melamed, 2000), to produce the final list of cognate pairs.

Note that the above-described cognate extraction algorithm has three important components: (1) orthographic, based on LCSR, (2) semantic, based on word alignments and pivoting over English, and (3) competitive linking. The orthographic component is essential when looking for cognates since they must have similar spelling by definition, while the semantic component prevents the extraction of false friends like вреден, which means 'valuable' in Macedonian but 'harmful' in Bulgarian. Finally, competitive linking helps prevent issues related to word inflection that cannot be handled using the semantic component alone.

3.2 Transliteration Training

For each pair in the list of cognate pairs, we added spaces between any two adjacent letters for both words, and we further appended special start and end characters. We split the resulting list into training, development and testing parts and we trained and tuned a character-level Macedonian-Bulgarian phrase-based monotone SMT system similar to that in (Finch and Sumita, 2008; Tiedemann and Nabende, 2009; Nakov and Ng, 2009; Nakov and Ng, 2012). The system used a character-level Bulgarian language model trained on words. We set the maximum phrase length and the language model order to 10, and we tuned the system using MERT.

3.3 Transliteration Lattice Generation

Given a Macedonian sentence, we generated a lattice where each input Macedonian word of length three or longer was augmented with Bulgarian alternatives: n-best transliterations generated by the above character-level Macedonian-Bulgarian SMT system (after the characters were concatenated to form a word and the special symbols were removed).

In the lattice, we assigned the original Macedonian word the weight of 1; for the alternatives, we assigned scores between 0 and 1 that were the sum of the translation model probabilities of generating each alternative (the sum was needed since some options appeared multiple times in the n-best list).

4 Experiments and Evaluation

For our experiments, we used translated movie subtitles from the OPUS corpus (Tiedemann, 2009b). For Macedonian-Bulgarian there were only about 102,000 aligned sentences containing approximately 1.3 million tokens altogether. There was substantially more monolingual data available for Bulgarian: about 16 million sentences containing ca. 136 million tokens.

However, this data was noisy. Thus, we realigned the corpus using `hunalign` and we removed some Bulgarian files that were misclassified as Macedonian and vice versa, using a BLEU-filter. Furthermore, we also removed sentence pairs containing language-specific characters on the wrong side. From the remaining data we selected 10,000 sentence pairs (roughly 128,000 words) for development and another 10,000 (ca. 125,000 words) for testing; we used the rest for training.

The evaluation results are summarized in Table 3.

MK→BG		BLEU %	NIST	TER	METEOR
Transliteration					
	no translit.	10.74	3.33	67.92	60.30
t1	letter-based	12.07	3.61	66.42	61.87
t2	cogn.+lattice	22.74	5.51	55.99	66.42
Word-level SMT					
w0	Apertium	21.28	5.27	56.92	66.35
w1	SMT baseline	**31.10**	6.56	50.72	70.53
w2	w1 + t1-lattice	**32.19**$^{(+1.19)}$	6.76	49.68	71.18
Character-level SMT					
c1	char-aligned	**32.28**$^{(+1.18)}$	6.70	49.70	71.35
c2	bigram-aligned	**32.71**$^{(+1.61)}$	6.77	49.23	71.65
	trigram-aligned	32.07$^{(+0.97)}$	6.68	49.82	71.21
System combination					
	w2 + c2	32.92$^{(+1.82)}$	6.90	48.73	71.71
	w1 + c2	33.31$^{(+2.21)}$	6.91	48.60	71.81
Merged phrase tables					
m1	w1 + c2	**33.33**$^{(+2.13)}$	6.86	48.86	71.73
m2	w2 + c2	**33.94**$^{(+2.84)}$	6.89	48.99	71.76

Table 3: **Macedonian-Bulgarian translation and transliteration.** Superscripts show the absolute improvement in BLEU compared to the word-level baseline (w1).

Transliteration. The top rows of Table 3 show the results for Macedonian-Bulgarian transliteration. First, we can see that the BLEU score for the original Macedonian testset evaluated against the Bulgarian reference is 10.74, which is quite high and reflects the similarity between the two languages. The next line (t1) shows that many differences between Macedonian and Bulgarian stem from mere differences in orthography: we mapped the six letters in the Macedonian alphabet that do not exist in the Bulgarian alphabet to corresponding Bulgarian letters and letter sequences, gaining over 1.3 BLEU points. The following line (t2) shows the results using the sophisticated transliteration described in Section 3, which takes two kinds of context into account: (1) word-internal letter context, and (2) sentence-level word context. We generated a lattice for each Macedonian test sentence, which included the original Macedonian words and the 1-best[2] Bulgarian transliteration option from the character-level transliteration model. We then decoded the lattice using a Bulgarian language model; this increased BLEU to 22.74.

Word-level translation. Naturally, lattice-based transliteration cannot really compete against standard word-level translation (w1), which is better by 8 BLEU points. Still, as line (w2) shows, using the 1-best transliteration lattice as an input to (w1) yields[3] consistent improvement over (w1) for four evaluation metrics: BLEU (Papineni et al., 2002), NIST v. 13, TER (Snover et al., 2006) v. 0.7.25, and METEOR (Lavie and Denkowski, 2009) v. 1.3. The baseline system is also significantly better than the on-line version of Apertium (http://www.apertium.org/), a shallow transfer-rule-based MT system that is optimized for closely-related languages (accessed on 2012/05/02). Here, Apertium suffers badly from a large number of unknown words in our testset (ca. 15%).

Character-level translation. Moving down to the next group of experiments in Table 3, we can see that standard character-level SMT (c1), i.e., simply treating characters as separate words, performs significantly better than word-level SMT. Using bigram-based character alignments yields further improvement of +0.43 BLEU.

System combination. Since word-level and character-level models have different strengths and weaknesses, we further tried to combine them. We used MEMT, a state-of-the-art Multi-Engine Machine Translation system (Heafield and Lavie, 2010), to combine the outputs of (c3) with the output of (w1) and of (w2). Both combinations improved over the individual systems, but (w1)+(c2) performed better, by +0.6 BLEU points over (c2).

Combining word-level and phrase-level SMT. Finally, we also combined (w1) with (c3) in a more direct way: by merging their phrase tables. First, we split the phrases in the word-level phrase tables of (w1) to characters as in character-level models. Then, we generated four versions of each phrase pair: with/without "_" at the beginning/end of the phrase. Finally, we merged these phrase pairs with those in the phrase table of (c3), adding two extra features indicating each phrase pair's origin: the first/second feature is 1 if the pair came from the first/second table, and 0.5 otherwise. This combination outperformed MEMT, probably because it expands the search space of the SMT system more directly. We further tried scoring with two language models in the process of translation, character-based and word-based, but we did not get consistent improvements. Finally, we experimented with a 1-best character-level lattice input that encodes the same options and weights as for (w2). This yielded our best overall BLEU score of 33.94, which is +2.84 BLEU points of absolute improvement over the (w1) baseline, and +1.23 BLEU points over (c2).[4]

5 Conclusion and Future Work

We have explored several combinations of character- and word-level translation models for translating between closely-related languages with scarce resources. In future work, we want to use such a model for pivot-based translations from the resource-poor language (Macedonian) to other languages (such as English) via the related language (Bulgarian).

Acknowledgments

The research is partially supported by the EU ICT PSP project LetsMT!, grant number 250456.

[2]Using 3/5/10/100-best made very little difference.

[3]The decoder can choose between (a) translating a Macedonian word and (b) using its 1-best Bulgarian transliteration.

[4]All improvements over (w1) in Table 3 that are greater or equal to 0.97 BLEU points are statistically significant according to Collins' sign test (Collins et al., 2005).

References

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

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of HLT-NAACL '06*, pages 17–24, New York, NY.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL '05*, pages 531–540, Ann Arbor, MI.

Robert Damper, Yannick Marchand, John-David Marsters, and Alex Bazin. 2005. Aligning text and phonemes for speech technology applications using an EM-like algorithm. *International Journal of Speech Technology*, 8(2):149–162.

Andrew Finch and Eiichiro Sumita. 2008. Phrase-based machine transliteration. In *Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation*, pages 13–18, Hyderabad, India.

Kenneth Heafield and Alon Lavie. 2010. Combining machine translation output with open source: The Carnegie Mellon multi-engine machine translation scheme. *The Prague Bulletin of Mathematical Linguistics*, 93(1):27–36.

Sittichai Jiampojamarn, Grzegorz Kondrak, and Tarek Sherif. 2007. Applying many-to-many alignments and hidden Markov models to letter-to-phoneme conversion. In *Proceedings of NAACL-HLT '07*, pages 372–379, Rochester, New York.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL '03*, pages 48–54, Edmonton, Canada.

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

Grzegorz Kondrak, Daniel Marcu, and Kevin Knight. 2003. Cognates can improve statistical translation models. In *Proceedings of NAACL '03*, pages 46–48, Edmonton, Canada.

Alon Lavie and Michael Denkowski. 2009. The Meteor metric for automatic evaluation of machine translation. *Machine Translation*, 23:105–115.

David Matthews. 2007. Machine transliteration of proper names. Master's thesis, School of Informatics, University of Edinburgh, Edinburgh, UK.

Dan Melamed. 2000. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–249.

Preslav Nakov and Hwee Tou Ng. 2009. Improved statistical machine translation for resource-poor languages using related resource-rich languages. In *Proceedings of EMNLP '09*, pages 1358–1367, Singapore.

Preslav Nakov and Hwee Tou Ng. 2012. Improving statistical machine translation for a resource-poor language using related resource-rich languages. *Journal of cial Intelligence Research*, 44.

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 *Proceedings of ACL '03*, 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 ACL '02*, pages 311–318, Philadelphia, PA.

Eric Ristad and Peter Yianilos. 1998. Learning string edit distance. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 20(5):522–532.

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 AMTA '06*, pages 223–231.

Jörg Tiedemann and Peter Nabende. 2009. Translating transliterations. *International Journal of Computing and ICT Research*, 3(1):33–41.

Jörg Tiedemann. 2009a. Character-based PSMT for closely related languages. In *Proceedings of EAMT '09*, pages 12–19, Barcelona, Spain.

Jörg Tiedemann. 2009b. 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.

Jörg Tiedemann. 2012. Character-based pivot translation for under-resourced languages and domains. In *Proceedings of EACL '12*, pages 141–151, Avignon, France.

Masao Utiyama and Hitoshi Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *Proceedings of NAACL-HLT '07*, pages 484–491, Rochester, NY.

David Vilar, Jan-Thorsten Peter, and Hermann Ney. 2007. Can we translate letters? In *Proceedings of WMT '07*, pages 33–39, Prague, Czech Republic.

Hua Wu and Haifeng Wang. 2007. Pivot language approach for phrase-based statistical machine translation. *Machine Translation*, 21(3):165–181.

Improving the IBM Alignment Models Using Variational Bayes

Darcey Riley and **Daniel Gildea**
Computer Science Dept.
University of Rochester
Rochester, NY 14627

Abstract

Bayesian approaches have been shown to reduce the amount of overfitting that occurs when running the EM algorithm, by placing prior probabilities on the model parameters. We apply one such Bayesian technique, variational Bayes, to the IBM models of word alignment for statistical machine translation. We show that using variational Bayes improves the performance of the widely used GIZA++ software, as well as improving the overall performance of the Moses machine translation system in terms of BLEU score.

1 Introduction

The IBM Models of word alignment (Brown et al., 1993), along with the Hidden Markov Model (HMM) (Vogel et al., 1996), serve as the starting point for most current state-of-the-art machine translation systems, both phrase-based and syntax-based (Koehn et al., 2007; Chiang, 2005; Galley et al., 2004).

Both the IBM Models and the HMM are trained using the EM algorithm (Dempster et al., 1977). Recently, Bayesian techniques have become widespread in applications of EM to natural language processing tasks, as a very general method of controlling overfitting. For instance, Johnson (2007) showed the benefits of such techniques when applied to HMMs for unsupervised part of speech tagging. In machine translation, Blunsom et al. (2008) and DeNero et al. (2008) use Bayesian techniques to learn bilingual phrase pairs. In this setting, which involves finding a segmentation of the input sentences into phrasal units, it is particularly important to control the tendency of EM to choose longer phrases, which explain the training data well but are unlikely to generalize.

However, most state-of-the-art machine translation systems today are built on the basis of word-level alignments of the type generated by GIZA++ from the IBM Models and the HMM. Overfitting is also a problem in this context, and improving these word alignment systems could be of broad utility in machine translation research.

Moore (2004) discusses details of how EM overfits the data when training IBM Model 1. He discovers that the EM algorithm is particularly susceptible to overfitting in the case of rare words, due to the "garbage collection" phenomenon. Suppose a sentence contains an English word e_1 that occurs nowhere else in the data, and its French translation f_1. Suppose that same sentence also contains a word e_2 which occurs frequently in the overall data but whose translation in this sentence, f_2, co-occurs with it infrequently. If the translation $t(f_2|e_2)$ occurs with probability 0.1, then the sentence will have a higher probability if EM assigns the rare word and its actual translation a probability of $t(f_1|e_1) = 0.5$, and assigns the rare word's translation to f_2 a probability of $t(f_2|e_1) = 0.5$, than if it assigns a probability of 1 to the correct translation $t(f_1|e_1)$. Moore suggests a number of solutions to this issue, including add-n smoothing and initializing the probabilities based on a heuristic rather than choosing uniform probabilities. When combined, his solutions cause a significant decrease in alignment error rate (AER). More recently, Mermer and Saraclar (2011) have added a Bayesian prior to IBM Model 1 using Gibbs sampling for inference, showing improvements in BLEU scores.

In this paper, we describe the results of incorpo-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 306–310,
Jeju, Republic of Korea, 8-14 July 2012.

rating variational Bayes (VB) into the widely used GIZA++ software for word alignment. We use VB both because it converges more quickly than Gibbs sampling, and because it can be applied in a fairly straightforward manner to all of the models implemented by GIZA++. In Section 2, we describe VB in more detail. In Section 3, we present results for VB for the various models, in terms of perplexity of held-out test data, alignment error rate (AER), and the BLEU scores which result from using our version of GIZA++ in the end-to-end phrase-based machine translation system Moses.

2 Variational Bayes and GIZA++

Beal (2003) gives a detailed derivation of a variational Bayesian algorithm for HMMs. The result is a very slight change to the M step of the original EM algorithm. During the M step of the original algorithm, the expected counts collected in the E step are normalized to give the new values of the parameters:

$$\theta_{x_i|y} = \frac{E[c(x_i|y)]}{\sum_j E[c(x_j|y)]} \quad (1)$$

The variational Bayesian M step performs an inexact normalization, where the resulting parameters will add up to less than one. It does this by passing the expected counts collected in the E step through the function $f(v) = \exp(\psi(v))$, where ψ is the digamma function, and α is the hyperparameter of the Dirichlet prior (Johnson, 2007):

$$\theta_{x_i|y} = \frac{f(E[c(x_i|y)] + \alpha)}{f(\sum_j (E[c(x_j|y)] + \alpha))} \quad (2)$$

This modified M step can be applied to any model which uses a multinomial distribution; for this reason, it works for the IBM Models as well as HMMs, and is thus what we use for GIZA++.

In practice, the digamma function has the effect of subtracting 0.5 from its argument. When α is set to a low value, this results in "anti-smoothing". For the translation probabilities, because about 0.5 is subtracted from the expected counts, small counts corresponding to rare co-occurrences of words will be penalized heavily, while larger counts will not be affected very much. Thus, low values of α cause the algorithm to favor words which co-occur frequently and to distrust words that co-occur rarely.

Sentence pair	count
e_2 f_3	9
e_2 f_2	2
e_1 e_2 f_1 f_2	1

Table 1: An example of data with rare words.

In this way, VB controls the overfitting that would otherwise occur with rare words. On the other hand, higher values of α can be chosen if smoothing is desired, for instance in the case of the alignment probabilities, which state how likely a word in position i of the English sentence is to align to a word in position j of the French sentence. For these probabilities, smoothing is important because we do not want to rule out any alignment altogether, no matter how infrequently it occurs in the data.

We implemented VB for the translation probabilities as well as for the position alignment probabilities of IBM Model 2. We discovered that adding VB for the translation probabilities improved the performance of the system. However, including VB for the alignment probabilities had relatively little effect, because the alignment table in its original form does some smoothing during normalization by interpolating the counts with a uniform distribution. Because VB can itself be a form of smoothing, the two versions of the code behave similarly. We did not experiment with VB for the distortion probabilities of the HMM or Models 3 and 4, as these distributions have fewer parameters and are likely to have reliable counts during EM. Thus, in Section 3, we present the results of using VB for the translation probabilities only.

3 Results

First, we ran our modified version of GIZA++ on a simple test case designed to be similar to the example from Moore (2004) discussed in Section 1. Our test case, shown in Table 1, had three different sentence pairs; we included nine instances of the first, two instances of the second, and one of the third.

Human intuition tells us that f_2 should translate to e_2 and f_1 should translate to e_1. However, the EM algorithm without VB prefers e_1 as the translation

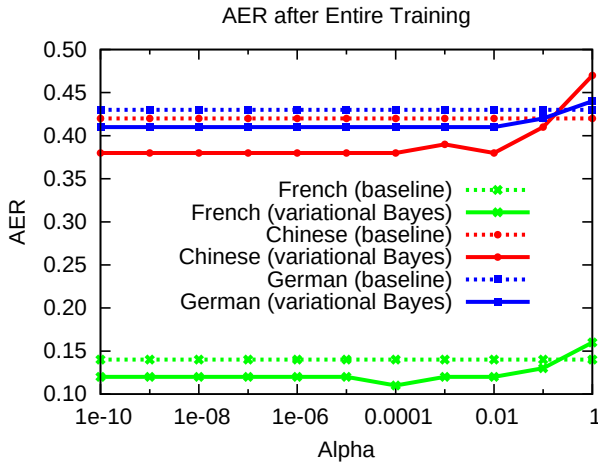


Figure 1: Determining the best value of α for the translation probabilities. Training data is 10,000 sentence pairs from each language pair. VB is used for Model 1 only. This table shows the AER for different values of α after training is complete (five iterations each of Models 1, HMM, 3, and 4).

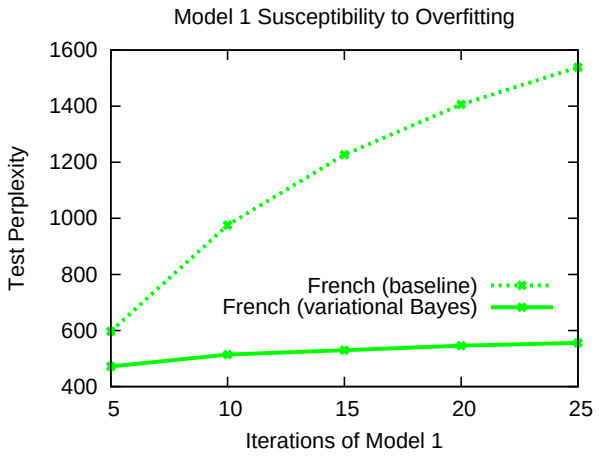


Figure 2: Effect of variational Bayes on overfitting for Model 1. Training data is 10,000 sentence pairs. This table contrasts the test perplexities of Model 1 with variational Bayes and Model 1 without variational Bayes after different numbers of training iterations. Variational Bayes successfully controls overfitting.

of f_2, due to the "garbage collection" phenomenon described above. The EM algorithm with VB does not overfit this data and prefers e_2 as f_2's translation.

For our experiments with bilingual data, we used three language pairs: French and English, Chinese and English, and German and English. We used Canadian Hansard data for French-English, Europarl data for German-English, and newswire data for Chinese-English. For measuring alignment error rate, we used 447 French-English sentences provided by Hermann Ney and Franz Och containing both sure and possible alignments, while for German-English we used 220 sentences provided by Chris Callison-Burch with sure alignments only, and for Chinese-English we used the first 400 sentences of the data provided by Yang Liu, also with sure alignments only. For computing BLEU scores, we used single reference datasets for French-English and German-English, and four references for Chinese-English. For minimum error rate training, we used 1000 sentences for French-English, 2000 sentences for German-English, and 1274 sentences for Chinese-English. Our test sets contained 1000 sentences each for French-English and German-English, and 686 sentences for Chinese-English. For scoring the Viterbi alignments of each system against gold-standard annotated alignments, we use the alignment error rate (AER) of Och and Ney (2000), which measures agreement at the level of pairs of words.

We ran our code on ten thousand sentence pairs to determine the best value of α for the translation probabilities $t(f|e)$. For our training, we ran GIZA++ for five iterations each of Model 1, the HMM, Model 3, and Model 4. Variational Bayes was only used for Model 1. Figure 1 shows how VB, and different values of α in particular, affect the performance of GIZA++ in terms of AER. We discover that, after all training is complete, VB improves the performance of the overall system, lowering AER (Figure 1) for all three language pairs. We find that low values of α cause the most consistent improvements, and so we use $\alpha = 0$ for the translation probabilities in the remaining experiments. Note that, while a value of $\alpha = 0$ does not define a probabilistically valid Dirichlet prior, it does not cause any practical problems in the update equation for VB.

Figure 2 shows the test perplexity after GIZA++ has been run for twenty-five iterations of Model 1: without VB, the test perplexity increases as training continues, but it remains stable when VB is used. Thus, VB eliminates the need for the early stopping that is often employed with GIZA++.

After choosing 0 as the best value of α for the

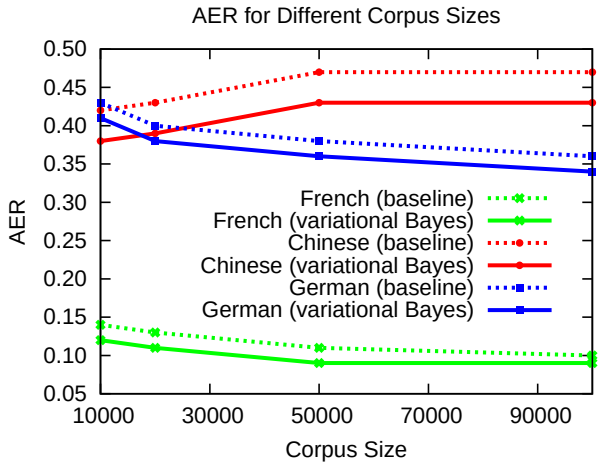


Figure 3: Performance of GIZA++ on different amounts of test data. Variational Bayes is used for Model 1 only. Table shows AER after all the training has completed (five iterations each of Models 1, HMM, 3, and 4).

	AER		
	French	Chinese	German
Baseline	0.14	0.42	0.43
M1 Only	0.12	0.39	0.41
HMM Only	0.14	0.42	0.42
M3 Only	0.14	0.42	0.43
M4 Only	0.14	0.42	0.43
All Models	0.19	0.44	0.45

Table 2: Effect of Adding Variational Bayes to Specific Models

translation probabilities, we reran the test above (five iterations each of Models 1, HMM, 3, and 4, with VB turned on for Model 1) on different amounts of data. We found that the results for larger data sizes were comparable to the results for ten thousand sentence pairs, both with and without VB (Figure 3).

We then tested whether VB should be used for the later models. In all of these experiments, we ran Models 1, HMM, 3, and 4 for five iterations each, training on the same ten thousand sentence pairs that we used in the previous experiments. In Table 2, we show the performance of the system when no VB is used, when it is used for each of the four models individually, and when it is used for all four models simultaneously. We saw the most overall improvement when VB was used only for Model 1; using VB for all four models simultaneously caused the most improvement to the test perplexity, but at the cost of the AER.

	BLEU Score		
	French	Chinese	German
Baseline	26.34	21.03	21.14
M1 Only	26.54	21.58	21.73
All Models	26.46	22.08	21.96

Table 3: BLEU Scores

For the MT experiments, we ran GIZA++ through Moses, training Model 1, the HMM, and Model 4 on 100,000 sentence pairs from each language pair. We ran three experiments, one with VB turned on for all models, one with VB turned on for Model 1 only, and one (the baseline) with VB turned off for all models. When VB was turned on, we ran GIZA++ for five iterations per model as in our earlier tests, but when VB was turned off, we ran GIZA++ for only four iterations per model, having determined that this was the optimal number of iterations for baseline system. VB was used for the translation probabilities only, with α set to 0.

As can be seen in Table 3, using VB increases the BLEU score for all three language pairs. For French, the best results were achieved when VB was used for Model 1 only; for Chinese and German, on the other hand, using VB for all models caused the most improvements. For French, the BLEU score increased by 0.20; for German, it increased by 0.82; for Chinese, it increased by 1.05. Overall, VB seems to have the greatest impact on the language pairs that are most difficult to align and translate to begin with.

4 Conclusion

We find that applying variational Bayes with a Dirichlet prior to the translation models implemented in GIZA++ improves alignments, both in terms of AER and the BLEU score of an end-to-end translation system. Variational Bayes is especially beneficial for IBM Model 1, because its lack of fertility and position information makes it particularly susceptible to the garbage collection phenomenon. Applying VB to Model 1 alone tends to improve the performance of later models in the training sequence. Model 1 is an essential stepping stone in avoiding local minima when training the following models, and improvements to Model 1 lead to improvements in the end-to-end system.

References

Matthew J. Beal. 2003. *Variational Algorithms for Approximate Bayesian Inference*. Ph.D. thesis, University College London.

Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. Bayesian synchronous grammar induction. In *Neural Information Processing Systems (NIPS)*.

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(2):263–311.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL-05*, pages 263–270, Ann Arbor, MI.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society*, 39(1):1–21.

John DeNero, Alexandre Bouchard-Côté, and Dan Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 314–323, Honolulu, Hawaii, October.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of NAACL-04*, pages 273–280, Boston.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 296–305, Prague, Czech Republic, June. 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 ACL, Demonstration Session*, pages 177–180.

Coskun Mermer and Murat Saraclar. 2011. Bayesian word alignment for statistical machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*, pages 182–187.

Robert C. Moore. 2004. Improving IBM word alignment Model 1. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 518–525, Barcelona, Spain, July.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL-00*, pages 440–447, Hong Kong, October.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING-96*, pages 836–841.

Post-ordering by Parsing for Japanese-English Statistical Machine Translation

Isao Goto **Masao Utiyama** **Eiichiro Sumita**
Multilingual Translation Laboratory, MASTAR Project
National Institute of Information and Communications Technology
3-5 Hikaridai, Keihanna Science City, Kyoto, 619-0289, Japan
{igoto, mutiyama, eiichiro.sumita}@nict.go.jp

Abstract

Reordering is a difficult task in translating between widely different languages such as Japanese and English. We employ the post-ordering framework proposed by (Sudoh et al., 2011b) for Japanese to English translation and improve upon the reordering method. The existing post-ordering method reorders a sequence of target language words in a source language word order via SMT, while our method reorders the sequence by: 1) parsing the sequence to obtain syntax structures similar to a source language structure, and 2) transferring the obtained syntax structures into the syntax structures of the target language.

1 Introduction

The word reordering problem is a challenging one when translating between languages with widely different word orders such as Japanese and English. Many reordering methods have been proposed in statistical machine translation (SMT) research. Those methods can be classified into the following three types:

Type-1: Conducting the target word selection and reordering jointly. These include phrase-based SMT (Koehn et al., 2003), hierarchical phrase-based SMT (Chiang, 2007), and syntax-based SMT (Galley et al., 2004; Ding and Palmer, 2005; Liu et al., 2006; Liu et al., 2009).

Type-2: Pre-ordering (Xia and McCord, 2004; Collins et al., 2005; Tromble and Eisner, 2009; Ge, 2010; Isozaki et al., 2010b; DeNero and Uszkoreit, 2011; Wu et al., 2011). First, these methods reorder the source language sentence into the target language word order. Then, they translate the reordered source word sequence using SMT methods.

Type-3: Post-ordering (Sudoh et al., 2011b; Matusov et al., 2005). First, these methods translate the source sentence almost monotonously into a sequence of the target language words. Then, they reorder the translated word sequence into the target language word order.

This paper employs the post-ordering framework for Japanese-English translation based on the discussions given in Section 2, and improves upon the reordering method. Our method uses syntactic structures, which are essential for improving the target word order in translating long sentences between Japanese (a Subject-Object-Verb (SOV) language) and English (an SVO language).

Before explaining our method, we explain the pre-ordering method for English to Japanese used in the post-ordering framework.

In English-Japanese translation, Isozaki et al. (2010b) proposed a simple pre-ordering method that achieved the best quality in human evaluations, which were conducted for the NTCIR-9 patent machine translation task (Sudoh et al., 2011a; Goto et al., 2011). The method, which is called *head finalization*, simply moves syntactic heads to the end of corresponding syntactic constituents (e.g., phrases and clauses). This method first changes the English word order into a word order similar to Japanese word order using the head finalization rule. Then, it translates (almost monotonously) the pre-ordered

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 311–316,
Jeju, Republic of Korea, 8-14 July 2012.

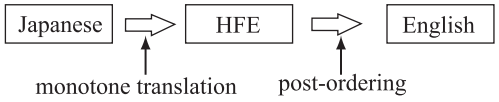


Figure 1: Post-ordering framework.

English words into Japanese.

There are two key reasons why this pre-ordering method works for estimating Japanese word order. The first reason is that Japanese is a typical head-final language. That is, a syntactic head word comes after nonhead (dependent) words. Second, input English sentences are parsed by a high-quality parser, Enju (Miyao and Tsujii, 2008), which outputs syntactic heads. Consequently, the parsed English input sentences can be pre-ordered into a Japanese-like word order using the head finalization rule.

Pre-ordering using the head finalization rule naturally cannot be applied to Japanese-English translation, because English is not a head-final language. If we want to pre-order Japanese sentences into an English-like word order, we therefore have to build complex rules (Sudoh et al., 2011b).

2 Post-ordering for Japanese to English

Sudoh et al. (2011b) proposed a post-ordering method for Japanese-English translation. The translation flow for the post-ordering method is shown in Figure 1, where "HFE" is an abbreviation of "Head Final English". An HFE sentence consists of English words in a Japanese-like structure. It can be constructed by applying the head-finalization rule (Isozaki et al., 2010b) to an English sentence parsed by Enju. Therefore, if good rules are applied to this HFE sentence, the underlying English sentence can be recovered. This is the key observation of the post-ordering method.

The process of post-ordering translation consists of two steps. First, the Japanese input sentence is translated into HFE almost monotonously. Then, the word order of HFE is changed into an English word order.

Training for the post-ordering method is conducted by first converting the English sentences in a Japanese-English parallel corpus into HFE sentences using the head-finalization rule. Next, a monotone phrase-based Japanese-HFE SMT model is built using the Japanese-HFE parallel corpus whose HFE was converted from English. Finally, an HFE-to-English word reordering model is built using the HFE-English parallel corpus.

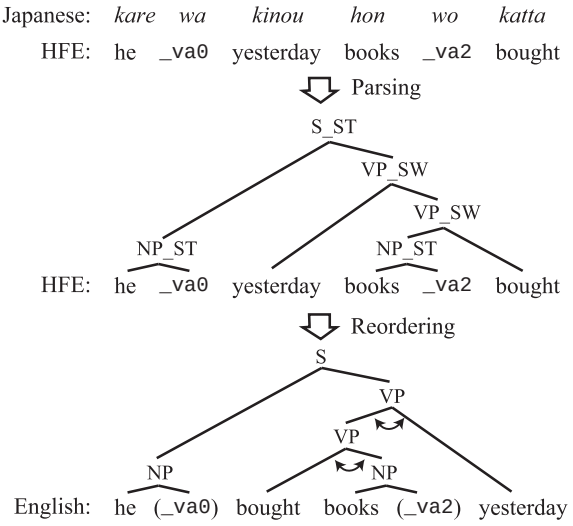


Figure 2: Example of post-ordering by parsing.

3 Post-ordering Models

3.1 SMT Model

Sudoh et al. (2011b) have proposed using phrase-based SMT for converting HFE sentences into English sentences. The advantage of their method is that they can use off-the-shelf SMT techniques for post-ordering.

3.2 Parsing Model

Our proposed model is called the *parsing model*. The translation process for the parsing model is shown in Figure 2. In this method, we first parse the HFE sentence into a binary tree. We then swap the nodes annotated with "_SW" suffixes in this binary tree in order to produce an English sentence.

The structures of the HFE sentences, which are used for training our parsing model, can be obtained from the corresponding English sentences as follows.[1] First, each English sentence in the training Japanese-English parallel corpus is parsed into a binary tree by applying Enju. Then, for each node in this English binary tree, the two children of each node are swapped if its first child is the head node (See (Isozaki et al., 2010b) for details of the head

[1]The explanations of pseudo-particles (_va0 and _va2) and other details of the HFE is given in Section 4.2.

final rules). At the same time, these swapped nodes are annotated with "_SW". When the two nodes are not swapped, they are annotated with "_ST" (indicating "Straight"). A node with only one child is not annotated with either "_ST" or "_SW". The result is an HFE sentence in a binary tree annotated with "_SW" and "_ST" suffixes.

Observe that the HFE sentences can be regarded as binary trees annotated with syntax tags augmented with swap/straight suffixes. Therefore, the structures of these binary trees can be learnable by using an off-the-shelf grammar learning algorithm. The learned parsing model can be regarded as an ITG model (Wu, 1997) between the HFE and English sentences. [2]

In this paper, we used the Berkeley Parser (Petrov and Klein, 2007) for learning these structures. The HFE sentences can be parsed by using the learned parsing model. Then the parsed structures can be converted into their corresponding English structures by swapping the "_SW" nodes. Note that this parsing model jointly learns how to parse and swap the HFE sentences.

4 Detailed Explanation of Our Method

This section explains the proposed method, which is based on the post-ordering framework using the parsing model.

4.1 Translation Method

First, we produce N-best HFE sentences using Japanese-to-HFE monotone phrase-based SMT. Next, we produce K-best parse trees for each HFE sentence by parsing, and produce English sentences by swapping any nodes annotated with "_SW". Then we score the English sentences and select the English sentence with the highest score.

For the score of an English sentence, we use the sum of the log-linear SMT model score for Japanese-to-HFE and the logarithm of the language model probability of the English sentence.

[2]There are works using the ITG model in SMT: ITG was used for training pre-ordering models (DeNero and Uszkoreit, 2011); hierarchical phrase-based SMT (Chiang, 2007), which is an extension of ITG; and reordering models using ITG (Chen et al., 2009; He et al., 2010). These methods are not post-ordering methods.

4.2 HFE and Articles

This section describes the details of HFE sentences. In HFE sentences: 1) Heads are final except for coordination. 2) Pseudo-particles are inserted after verb arguments: `_va0` (subject of sentence head), `_va1` (subject of verb), and `_va2` (object of verb). 3) Articles (a, an, the) are dropped.

In our method of HFE construction, unlike that used by (Sudoh et al., 2011b), plural nouns are left as-is instead of converted to the singular.

Applying our parsing model to an HFE sentence produces an English sentence that does not have articles, but does have pseudo-particles. We removed the pseudo-particles from the reordered sentences before calculating the probabilities used for the scores of the reordered sentences. A reordered sentence without pseudo-particles is represented by E. A language model $P(E)$ was trained from English sentences whose articles were dropped.

In order to output a genuine English sentence E' from E, articles must be inserted into E. A language model trained using genuine English sentences is used for this purpose. We try to insert one of the articles $\{$a, an, the$\}$ or no article for each word in E. Then we calculate the maximum probability word sequence through dynamic programming for obtaining E'.

5 Experiment

5.1 Setup

We used patent sentence data for the Japanese to English translation subtask from the NTCIR-9 and 8 (Goto et al., 2011; Fujii et al., 2010). There were 2,000 test sentences for NTCIR-9 and 1,251 for NTCIR-8. XML entities included in the data were decoded to UTF-8 characters before use.

We used Enju (Miyao and Tsujii, 2008) v2.4.2 for parsing the English side of the training data. Mecab [3] v0.98 was used for the Japanese morphological analysis. The translation model was trained using sentences of 64 words or less from the training corpus as (Sudoh et al., 2011b). We used 5-gram language models using SRILM (Stolcke et al., 2011).

We used the Berkeley parser (Petrov and Klein, 2007) to train the parsing model for HFE and to

[3]http://mecab.sourceforge.net/

parse HFE. The parsing model was trained using 0.5 million sentences randomly selected from training sentences of 40 words or less. We used the phrase-based SMT system Moses (Koehn et al., 2007) to calculate the SMT score and to produce HFE sentences. The distortion limit was set to 0. We used 10-best Moses outputs and 10-best parsing results of Berkeley parser.

5.2 Compared Methods

We used the following 5 comparison methods: Phrase-based SMT (PBMT), Hierarchical phrase-based SMT (HPBMT), String-to-tree syntax-based SMT (SBMT), Post-ordering based on phrase-based SMT (PO-PBMT) (Sudoh et al., 2011b), and Post-ordering based on hierarchical phrase-based SMT (PO-HPBMT).

We used Moses for these 5 systems. For PO-PBMT, a distortion limit 0 was used for the Japanese-to-HFE translation and a distortion limit 20 was used for the HFE-to-English translation. The PO-HPBMT method changes the post-ordering method of PO-PBMT from a phrase-based SMT to a hierarchical phrase-based SMT. We used a max-chart-span 15 for the hierarchical phrase-based SMT. We used distortion limits of 12 or 20 for PBMT and a max-chart-span 15 for HPBMT.

The parameters for SMT were tuned by MERT using the first half of the development data with HFE converted from English.

5.3 Results and Discussion

We evaluated translation quality based on the case-insensitive automatic evaluation scores of RIBES v1.1 (Isozaki et al., 2010a) and BLEU-4. The results are shown in Table 1.

Ja-to-En	NTCIR-9		NTCIR-8	
	RIBES	BLEU	RIBES	BLEU
Proposed	**72.57**	**31.75**	**73.48**	**32.80**
PBMT (limit 12)	68.44	29.64	69.18	30.72
PBMT (limit 20)	68.86	30.13	69.63	31.22
HPBMT	69.92	30.15	70.18	30.94
SBMT	69.22	29.53	69.87	30.37
PO-PBMT	68.81	30.39	69.80	31.71
PO-HPBMT	70.47	27.49	71.34	28.78

Table 1: Evaluation results (case insensitive).

From the results, the proposed method achieved the best scores for both RIBES and BLEU for NTCIR-9 and NTCIR-8 test data. Since RIBES is sensitive to global word order and BLEU is sensitive to local word order, the effectiveness of the proposed method for both global and local reordering can be demonstrated through these comparisons.

In order to investigate the effects of our post-ordering method in detail, we conducted an "HFE-to-English reordering" experiment, which shows the main contribution of our post-ordering method in the framework of post-ordering SMT as compared with (Sudoh et al., 2011b). In this experiment, we changed the word order of the oracle-HFE sentences made from reference sentences into English, this is the same way as Table 4 in (Sudoh et al., 2011b). The results are shown in Table 2.

This results show that our post-ordering method is more effective than PO-PBMT and PO-HPBMT. Since RIBES is based on the rank order correlation coefficient, these results show that the proposed method correctly recovered the word order of the English sentences. These high scores also indicate that the parsing results for high quality HFE are fairly trustworthy.

oracle-HFE-to-En	NTCIR-9		NTCIR-8	
	RIBES	BLEU	RIBES	BLEU
Proposed	**94.66**	**80.02**	**94.93**	**79.99**
PO-PBMT	77.34	62.24	78.14	63.14
PO-HPBMT	77.99	53.62	80.85	58.34

Table 2: Evaluation resutls focusing on post-ordering.

In these experiments, we did not compare our method to pre-ordering methods. However, some groups used pre-ordering methods in the NTCIR-9 Japanese to English translation subtask. The NTT-UT (Sudoh et al., 2011a) and NAIST (Kondo et al., 2011) groups used pre-ordering methods, but could not produce RIBES and BLEU scores that both were better than those of the baseline results. In contrast, our method was able to do so.

6 Conclusion

This paper has described a new post-ordering method. The proposed method parses sentences that consist of target language words in a source language word order, and does reordering by transferring the syntactic structures similar to the source language syntactic structures into the target language syntactic structures.

References

Han-Bin Chen, Jian-Cheng Wu, and Jason S. Chang. 2009. Learning Bilingual Linguistic Reordering Model for Statistical Machine Translation. In *Proceedings of Human Language Technologies: The 2009 NAACL*, pages 254–262, Boulder, Colorado, June. Association for Computational Linguistics.

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause Restructuring for Statistical Machine Translation. In *Proceedings of the 43rd ACL*, pages 531–540, Ann Arbor, Michigan, June. Association for Computational Linguistics.

John DeNero and Jakob Uszkoreit. 2011. Inducing Sentence Structure from Parallel Corpora for Reordering. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Yuan Ding and Martha Palmer. 2005. Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars. In *Proceedings of the 43rd ACL*, pages 541–548, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Atsushi Fujii, Masao Utiyama, Mikio Yamamoto, Takehito Utsuro, Terumasa Ehara, Hiroshi Echizen-ya, and Sayori Shimohata. 2010. Overview of the Patent Translation Task at the NTCIR-8 Workshop. In *Proceedings of NTCIR-8*, pages 371–376.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Niyu Ge. 2010. A Direct Syntax-Driven Reordering Model for Phrase-Based Machine Translation. In *Proceedings of NAACL-HLT*, pages 849–857, Los Angeles, California, June. Association for Computational Linguistics.

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 NTCIR-9*, pages 559–578.

Yanqing He, Yu Zhou, Chengqing Zong, and Huilin Wang. 2010. A Novel Reordering Model Based on Multi-layer Phrase for Statistical Machine Translation. In *Proceedings of the 23rd Coling*, pages 447–455, Beijing, China, August. Coling 2010 Organizing Committee.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010a. Automatic Evaluation of Translation Quality for Distant Language Pairs. In *Proceedings of the 2010 EMNLP*, pages 944–952.

Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010b. Head Finalization: A Simple Reordering Rule for SOV Languages. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 244–251, Uppsala, Sweden, July. Association for Computational Linguistics.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the 2003 HLT-NAACL*, pages 48–54.

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 ACL*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

Shuhei Kondo, Mamoru Komachi, Yuji Matsumoto, Katsuhito Sudoh, Kevin Duh, and Hajime Tsukada. 2011. Learning of Linear Ordering Problems and its Application to J-E Patent Translation in NTCIR-9 PatentMT. In *Proceedings of NTCIR-9*, pages 641–645.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In *Proceedings of the 21st ACL*, pages 609–616, Sydney, Australia, July. Association for Computational Linguistics.

Yang Liu, Yajuan Lü, and Qun Liu. 2009. Improving Tree-to-Tree Translation with Packed Forests. 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 558–566, Suntec, Singapore, August. Association for Computational Linguistics.

E. Matusov, S. Kanthak, and Hermann Ney. 2005. On the Integration of Speech Recognition and Statistical Machine Translation. In *Proceedings of Interspeech*, pages 3177–3180.

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature Forest Models for Probabilistic HPSG Parsing. In *Computational Linguistics, Volume 34, Number 1*, pages 81–88.

Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. In *NAACL-HLT*, pages 404–411, Rochester, New York, April. Association for Computational Linguistics.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. SRILM at Sixteen: Update and Outlook. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop*.

Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, Masaaki Nagata, Xianchao Wu, Takuya Matsuzaki, and Jun'ichi Tsujii. 2011a. NTT-UT Statistical Machine Translation in NTCIR-9 PatentMT. In *Proceedings of NTCIR-9*, pages 585–592.

Katsuhito Sudoh, Xianchao Wu, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011b. Post-ordering in Statistical Machine Translation. In *Proceedings of the 13th Machine Translation Summit*, pages 316–323.

Roy Tromble and Jason Eisner. 2009. Learning Linear Ordering Problems for Better Translation. In *Proceedings of the 2009 EMNLP*, pages 1007–1016, Singapore, August. Association for Computational Linguistics.

Xianchao Wu, Katsuhito Sudoh, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Extracting Preordering Rules from Chunk-based Dependency Trees for Japanese-to-English Translation. In *Proceedings of the 13th Machine Translation Summit*, pages 300–307.

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*, pages 508–514, Geneva, Switzerland, Aug 23–Aug 27. COLING.

An Exploration of Forest-to-String Translation: Does Translation Help or Hurt Parsing?

Hui Zhang
University of Southern California
Department of Computer Science
hzhang@isi.edu

David Chiang
University of Southern California
Information Sciences Institute
chiang@isi.edu

Abstract

Syntax-based translation models that operate on the output of a source-language parser have been shown to perform better if allowed to choose from a set of possible parses. In this paper, we investigate whether this is because it allows the translation stage to overcome parser errors or to override the syntactic structure itself. We find that it is primarily the latter, but that under the right conditions, the translation stage does correct parser errors, improving parsing accuracy on the Chinese Treebank.

1 Introduction

Tree-to-string translation systems (Liu et al., 2006; Huang et al., 2006) typically employ a pipeline of two stages: a syntactic parser for the source language, and a decoder that translates source-language trees into target-language strings. Originally, the output of the parser stage was a single parse tree, and this type of system has been shown to outperform phrase-based translation on, for instance, Chinese-to-English translation (Liu et al., 2006). More recent work has shown that translation quality is improved further if the parser outputs a weighted parse *forest*, that is, a representation of a whole distribution over possible parse trees (Mi et al., 2008). In this paper, we investigate two hypotheses to explain why.

One hypothesis is that *forest-to-string translation selects worse parses*. Although syntax often helps translation, there may be situations where syntax, or at least syntax in the way that our models use it, can impose constraints that are too rigid for good-quality translation (Liu et al., 2007; Zhang et al., 2008). For example, suppose that a tree-to-string system encounters the following correct tree (only partial bracketing shown):

(1) [$_{\text{NP}}$ jīngjì zēngzhǎng] de sùdù
economy growth DE rate
'economic growth rate'

Suppose further that the model has never seen this phrase before, although it has seen the subphrase *zēngzhǎng de sùdù* 'growth rate'. Because this subphrase is not a syntactic unit in sentence (1), the system will be unable to translate it. But a forest-to-string system would be free to choose another (incorrect but plausible) bracketing:

(2) jīngjì [$_{\text{NP}}$ zēngzhǎng de sùdù]
economy growth DE rate

and successfully translate it using rules learned from observed data.

The other hypothesis is that *forest-to-string translation selects better parses*. For example, if a Chinese parser is given the input *cānjiā biǎojiě de hūnlǐ*, it might consider two structures:

(3) [$_{\text{VP}}$ cānjiā biǎojiě] de hūnlǐ
attend cousin DE wedding
'wedding that attends a cousin'

(4) cānjiā [$_{\text{NP}}$ biǎojiě de hūnlǐ]
attend cousin DE wedding
'attend a cousin's wedding'

The two structures have two different translations into English, shown above. While the parser prefers structure (3), an *n*-gram language model would easily prefer translation (4) and, therefore, its corresponding Chinese parse.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 317–321,
Jeju, Republic of Korea, 8-14 July 2012.

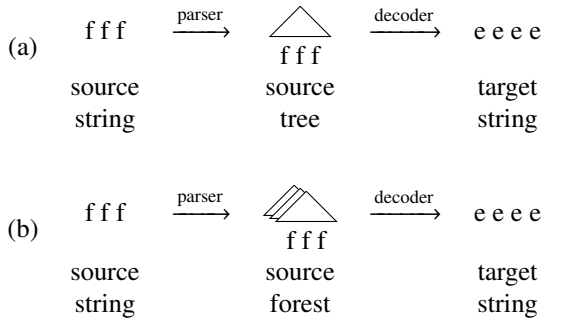


Figure 1: (a) In tree-to-string translation, the parser generates a single tree which the decoder must use to generate a translation. (b) In forest-to-string translation, the parser generates a forest of possible trees, any of which the decoder can use to generate a translation.

Previous work has shown that an observed target-language translation can improve parsing of source-language text (Burkett and Klein, 2008; Huang et al., 2009), but to our knowledge, only Chen et al. (2011) have explored the case where the target-language translation is unobserved.

Below, we carry out experiments to test these two hypotheses. We measure the accuracy (using labeled-bracket F1) of the parses that the translation model selects, and find that they are worse than the parses selected by the parser. Our basic conclusion, then, is that the parses that help translation (according to B) are, on average, worse parses. That is, forest-to-string translation hurts parsing.

But there is a twist. Neither labeled-bracket F1 nor B is a perfect metric of the phenomena it is meant to measure, and our translation system is optimized to maximize B . If we optimize our system to maximize labeled-bracket F1 instead, we find that our translation system selects parses that score higher than the baseline parser's. That is, forest-to-string translation can help parsing.

2 Background

We provide here only a cursory overview of tree-to-string and forest-to-string translation. For more details, the reader is referred to the original papers describing them (Liu et al., 2006; Mi et al., 2008).

Figure 1a illustrates the tree-to-string translation pipeline. The parser stage can be any phrase-structure parser; it computes a parse for each source-language string. The decoder stage translates the source-language tree into a target-language string, using a synchronous tree-substitution grammar.

In forest-to-string translation (Figure 1b), the parser outputs a forest of possible parses of each source-language string. The decoder uses the same rules as in tree-to-string translation, but is free to select any of the trees contained in the parse forest.

3 Translation hurts parsing

The simplest experiment to carry out is to examine the parses actually selected by the decoder, and see whether they are better or worse than the parses selected by the parser. If they are worse, this supports the hypothesis that syntax can hurt translation. If they are better, we can conclude that translation can help parsing. In this initial experiment, we find that the former is the case.

3.1 Setup

The baseline parser is the Charniak parser (Charniak, 2000). We trained it on the Chinese Treebank (CTB) 5.1, split as shown in Table 1, following Duan et al. (2007).[1] The parser outputs a parse forest annotated with head words and other information. Since the decoder does not use these annotations, we use the max-rule algorithm (Petrov et al., 2006) to (approximately) sum them out. As a side benefit, this improves parsing accuracy from 77.76% to 78.42% F1. The weight of a hyperedge in this forest is its posterior probability, given the input string. We retain these weights as a feature in the translation model.

The decoder stage is a forest-to-string system (Liu et al., 2006; Mi et al., 2008) for Chinese-to-English translation. The datasets used are listed in Table 1. We generated word alignments with GIZA++ and symmetrized them using the *grow-diag-final-and* heuristic. We parsed the Chinese side using the Charniak parser as described above, and performed forest-based rule extraction (Mi and Huang, 2008) with a maximum height of 3 nodes. We used the same features as Mi and Huang (2008). The language model was a trigram model with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998), trained on the target

[1] The more common split, used by Bikel and Chiang (2000), has flaws that are described by Levy and Manning (2003).

	Parsing	Translation
Train	CTB 1–815 CTB 1101–1136	FBIS
Dev	CTB 900–931 CTB 1148–1151	NIST 2002
Test	CTB 816–885 CTB 1137–1147	NIST 2003

Table 1: Data used for training and testing the parsing and translation models.

System	Objective	Parsing F1%	Translation B %
Charniak	n/a	78.42	n/a
tree-to-string	max-B	78.42	23.07
forest-to-string	max-B	77.75	24.60
forest-to-string	max-F1	78.81	19.18

Table 2: Forest-to-string translation outperforms tree-to-string translation according to B , but the decreases parsing accuracy according to labeled-bracket F1. However, when we train to maximize labeled-bracket F1, forest-to-string translation yields better parses than both tree-to-string translation and the original parser.

side of the training data. We used minimum-error-rate (MER) training to optimize the feature weights (Och, 2003) to maximize B .

At decoding time, we select the best derivation and extract its source tree. In principle, we ought to sum over all derivations for each source tree; but the approximations that we tried (*n*-best list crunching, max-rule decoding, minimum Bayes risk) did not appear to help.

3.2 Results

Table 2 shows the main results of our experiments. In the second and third line, we see that the forest-to-string system outperforms the tree-to-string system by 1.53 B , consistent with previously published results (Mi et al., 2008; Zhang et al., 2009). However, we also find that the trees selected by the forest-to-string system score much lower according to labeled-bracket F1. This suggests that the reason the forest-to-string system is able to generate better translations is that it can soften the constraints imposed by the syntax of the source language.

4 Translation helps parsing

We have found that better translations can be obtained by settling for worse parses. However, translation accuracy is measured using B and parsing accuracy is measured using labeled-bracket F1, and neither of these is a perfect metric of the phenomenon it is meant to measure. Moreover, we optimized the translation model in order to maximize B . It is known that when MER training is used to optimize one translation metric, other translation metrics suffer (Och, 2003); much more, then, can we expect that optimizing B will cause labeled-bracket F1 to suffer. In this section, we try optimizing labeled-bracket F1, and find that, in this case, the translation model does indeed select parses that are better on average.

4.1 Setup

MER training with labeled-bracket F1 as an objective function is straightforward. At each iteration of MER training, we run the parser and decoder over the CTB dev set to generate an *n*-best list of possible translation derivations (Huang and Chiang, 2005). For each derivation, we extract its Chinese parse tree and compute the number of brackets guessed and the number matched against the gold-standard parse tree. A trivial modification of the MER trainer then optimizes the feature weights to maximize labeled-bracket F1.

A technical challenge that arises is ensuring diversity in the *n*-best lists. The MER trainer requires that each list contain enough unique translations (when maximizing B) or source trees (when maximizing labeled-bracket F1). However, because one source tree may lead to many translation derivations, the *n*-best list may contain only a few unique source trees, or in the extreme case, the derivations may all have the same source tree. We use a variant of the *n*-best algorithm that allows efficient generation of equivalence classes of derivations (Huang et al., 2006). The standard algorithm works by generating, at each node of the forest, a list of the best subderivations at that node; the variant drops a subderivation if it has the same source tree as a higher-scoring subderivation.

Maximum rule height	F1%
3	78.81
4	78.93
5	79.14

(a)

LM data (lines)	F1%
none	78.78
100	78.79
30k	78.67
300k	79.14
13M	79.24

(b)

Features	F1%
monolingual	78.89
+ bilingual	79.24

(c)

Parallel data (lines)	F1%
60k	78.00
120k	78.16
300k	79.24

(d)

Table 3: Effect of variations on parsing performance. (a) Increasing the maximum translation rule height increases parsing accuracy further. (b) Increasing/decreasing the language model size increases/decreases parsing accuracy. (c) Decreasing the parallel text size decreases parsing accuracy. (d) Removing all bilingual features decreases parsing accuracy, but only slightly.

4.2 Results

The last line of Table 2 shows the results of this second experiment. The system trained to optimize labeled-bracket F1 (*max-F1*) obtains a much lower B score than the one trained to maximize B (*max*-B)—unsurprisingly, because a single source-side parse can yield many different translations, but the objective function scores them equally. What is more interesting is that the max-F1 system obtains a higher F1 score, not only compared with the max-B system but also the original parser.

We then tried various settings to investigate what factors affect parsing performance. First, we found that increasing the maximum rule height increases F1 further (Table 3a).

One of the motivations of our method is that bilingual information (especially the language model) can help disambiguate the source side structures. To test this, we varied the size of the corpus used to train the language model (keeping a maximum rule height of 5 from the previous experiment). The 13M-line language model adds the Xinhua portion of Gigaword 3. In Table 3b we see that the parsing performance does increase with the language model size, with the largest language model yielding a net improvement of 0.82 over the baseline parser.

To test further the importance of bilingual information, we compared against a system built only from the Chinese side of the parallel text (with each word aligned to itself). We removed all features that use bilingual information, retaining only the parser probability and the phrase penalty. In their place we added a new feature, the probability of a rule's source side tree given its root label, which is essentially the same model used in Data-Oriented Parsing (Bod, 1992). Table 3c shows that this system still outperforms the original parser. In other words, part of the gain is not attributable to translation, but additional source-side context and data that the translation model happens to capture.

Finally, we varied the size of the parallel text (keeping a maximum rule height of 5 and the largest language model) and found that, as expected, parsing performance correlates with parallel data size (Table 3d).

5 Conclusion

We set out to investigate why forest-to-string translation outperforms tree-to-string translation. By comparing their performance as Chinese parsers, we found that forest-to-string translation sacrifices parsing accuracy, suggesting that forest-to-string translation works by overriding constraints imposed by syntax. But when we optimized the system to maximize labeled-bracket F1, we found that, in fact, forest-to-string translation is able to achieve higher accuracy, by 0.82 F1%, than the baseline Chinese parser, demonstrating that, to a certain extent, forest-to-string translation is able to correct parsing errors.

Acknowledgements

We are grateful to the anonymous reviewers for their helpful comments. This research was supported in part by DARPA under contract DOI-NBC D11AP00244.

References

Daniel M. Bikel and David Chiang. 2000. Two statistical parsing models applied to the Chinese Treebank. In *Proc. Second Chinese Language Processing Workshop*, pages 1–6.

Rens Bod. 1992. A computational model of language performance: Data Oriented Parsing. In *Proc. COLING 1992*, pages 855–859.

David Burkett and Dan Klein. 2008. Two languages are better than one (for syntactic parsing). In *Proc. EMNLP 2008*, pages 877–886.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. NAACL*, pages 132–139.

Stanley F. Chen and Joshua Goodman. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology.

Wenliang Chen, Jun'ichi Kazama, Min Zhang, Yoshimasa Tsuruoka, Yujie Zhang, Yiou Wang, Kentaro Torisawa, and Haizhou Li. 2011. SMT helps bitext dependency parsing. In *Proc. EMNLP 2011*, pages 73–83.

Xiangyu Duan, Jun Zhao, and Bo Xu. 2007. Probabilistic models for action-based Chinese dependency parsing. In *Proc. ECML 2007*, pages 559–566.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proc. IWPT 2005*, pages 53–64.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. AMTA 2006*, pages 65–73.

Liang Huang, Wenbin Jiang, and Qun Liu. 2009. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proc. EMNLP 2009*, pages 1222–1231.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for M-gram language modeling. In *Proc. ICASSP 1995*, pages 181–184.

Roger Levy and Christopher D. Manning. 2003. Is it harder to parse Chinese, or the Chinese Treebank? In *Proc. ACL 2003*, pages 439–446.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. COLING-ACL 2006*, pages 609–616.

Yang Liu, Yun Huang, Qun Liu, and Shouxun Lin. 2007. Forest-to-string statistical translation rules. In *Proc. ACL 2007*, pages 704–711.

Haitao Mi and Liang Huang. 2008. Forest-based translation rule extraction. In *Proc. EMNLP 2008*, pages 206–214.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. ACL-08: HLT*, pages 192–199.

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

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. COLING-ACL 2006*, pages 433–440.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan, and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *Proc. ACL-08: HLT*, pages 559–567.

Hui Zhang, Min Zhang, Haizhou Li, Aiti Aw, and Chew Lim Tan. 2009. Forest-based tree sequence to string translation model. In *Proc. ACL-IJCNLP 2009*, pages 172–180.

Unsupervised Morphology Rivals Supervised Morphology for Arabic MT

David Stallard **Jacob Devlin**
Michael Kayser
BBN Technologies
{stallard,jdevlin,rzbib}@bbn.com

Yoong Keok Lee **Regina Barzilay**
CSAIL
Massachusetts Institute of Technology
{yklee,regina}@csail.mit.edu

Abstract

If unsupervised morphological analyzers could approach the effectiveness of supervised ones, they would be a very attractive choice for improving MT performance on low-resource inflected languages. In this paper, we compare performance gains for state-of-the-art supervised vs. unsupervised morphological analyzers, using a state-of-the-art Arabic-to-English MT system. We apply maximum marginal decoding to the unsupervised analyzer, and show that this yields the best published segmentation accuracy for Arabic, while also making segmentation output more stable. Our approach gives an 18% relative BLEU gain for Levantine dialectal Arabic. Furthermore, it gives higher gains for Modern Standard Arabic (MSA), as measured on NIST MT-08, than does MADA (Habash and Rambow, 2005), a leading *supervised* MSA segmenter.

1 Introduction

If unsupervised morphological segmenters could approach the effectiveness of supervised ones, they would be a very attractive choice for improving machine translation (MT) performance in low-resource inflected languages. An example of particular current interest is Arabic, whose various colloquial dialects are sufficiently different from Modern Standard Arabic (MSA) in lexicon, orthography, and morphology, as to be low-resource languages themselves. An additional advantage of Arabic for study is the availability of high-quality supervised segmenters for MSA, such as MADA (Habash and Rambow, 2005), for performance comparison. The MT gain for supervised MSA segmenters on dialect establishes a lower bound, which the unsupervised segmenter must exceed if it is to be useful for dialect. And comparing the gain for supervised and unsupervised segmenters on MSA tells us how useful the unsupervised segmenter is, relative to the ideal case in which a supervised segmenter is available.

In this paper, we show that an unsupervised segmenter can in fact rival or surpass supervised MSA segmenters on MSA itself, while at the same time providing superior performance on dialect. Specifically, we compare the state-of-the-art morphological analyzer of Lee et al. (2011) with two leading supervised analyzers for MSA, MADA and Sakhr[1], each serving as an alternative preprocessor for a state-of-the-art statistical MT system (Shen et al., 2008). We measure MSA performance on NIST MT-08 (NIST, 2010), and dialect performance on a Levantine dialect web corpus (Zbib et al., 2012b).

To improve performance, we apply maximum marginal decoding (Johnson and Goldwater, 2009) (MM) to combine multiple runs of the Lee segmenter, and show that this dramatically reduces the variance and noise in the segmenter output, while yielding an improved segmentation accuracy that exceeds the best published scores for unsupervised segmentation on Arabic Treebank (Naradowsky and Toutanova, 2011). We also show that it yields MT-08 BLEU scores that are higher than those obtained with MADA, a leading *supervised* MSA segmenter. For Levantine, the segmenter increases BLEU score by 18% over the unsegmented baseline.

[1] http://www.sakhr.com/Default.aspx

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 322–327,
Jeju, Republic of Korea, 8-14 July 2012.

2 Related Work

Machine translation systems that process highly inflected languages often incorporate morphological analysis. Some of these approaches rely on morphological analysis for pre- and post-processing, while others modify the core of a translation system to incorporate morphological information (Habash, 2008; Luong et al., 2010; Nakov and Ng, 2011). For instance, factored translation Models (Koehn and Hoang, 2007; Yang and Kirchhoff, 2006; Avramidis and Koehn, 2008) parametrize translation probabilities as factors encoding morphological features.

The approach we have taken in this paper is an instance of a segmented MT model, which divides the input into morphemes and uses the derived morphemes as a unit of translation (Sadat and Habash, 2006; Badr et al., 2008; Clifton and Sarkar, 2011). This is a mainstream architecture that has been shown to be effective when translating from a morphologically rich language.

A number of recent approaches have explored the use of unsupervised morphological analyzers for MT (Virpioja et al., 2007; Creutz and Lagus, 2007; Clifton and Sarkar, 2011; Mermer and Akın, 2010; Mermer and Saraclar, 2011). Virpioja et al. (2007) apply the unsupervised morphological segmenter Morfessor (Creutz and Lagus, 2007), and apply an existing MT system at the level of morphemes. The system does not outperform the word baseline partially due to the insufficient accuracy of the automatic morphological analyzer.

The work of Mermer and Akın (2010) and Mermer and Saraclar (2011) attempts to integrate morphology and MT more closely than we do, by incorporating bilingual alignment probabilities into a Gibbs-sampled version of Morfessor for Turkish-to-English MT. However, the bilingual strategy shows no gain over the monolingual version, and neither version is competitive for MT with a supervised Turkish morphological segmenter (Oflazer, 1993). By contrast, the unsupervised analyzer we report on here yields MSA-to-English MT performance that equals or exceed the performance obtained with a leading supervised MSA segmenter, MADA (Habash and Rambow, 2005).

3 Review of Lee Unsupervised Segmenter

The segmenter of Lee et al. (2011) is a probabilistic model operating at word-type level. It is divided into four sub-model levels. **Model 1** prefers small affix lexicons, and assumes that morphemes are drawn independently. **Model 2** generates a latent POS tag for each word type, conditioning the word's affixes on the tag, thereby encouraging compatible affixes to be generated together. **Model 3** incorporates token-level contextual information, by generating word tokens with a type-level Hidden Markov Model (HMM). Finally, **Model 4** models morphosyntactic agreement with a transition probability distribution, encouraging adjacent tokens with the same endings to also have the same final suffix.

4 Applying Maximum Marginal Decoding to Reduce Variance and Noise

Maximum marginal decoding (Johnson and Goldwater, 2009) (MM) is a technique which assigns to each latent variable the value with the highest marginal probability, thereby maximizing the expected number of correct assignments (Rabiner, 1989). Johnson and Goldwater (2009) extend MM to Gibbs sampling by drawing a set of N independent Gibbs samples, and selecting for each word the most frequent segmentation found in them. They found that MM improved segmentation accuracy over the mean, consistent with its maximization criterion. However, for our setting, we find that MM provides several other crucial advantages as well.

First, MM dramatically reduces the output variance of Gibbs sampling (GS). Table 1 documents the severity of this variance for the MT-08 lexicon, as measured by the average exact-match accuracy and segmentation F-measure between different runs. It shows that on average, 13% of the word tokens, and 25% of the word types, are segmented differently from run to run, which obviously makes the input to MT highly unstable. By contrast the "MM" column of Table 1 shows that two different runs of MM, each derived by combining separate sets of 25 GS runs, agree on the segmentations of over 95% of the word token – a dramatic improvement in stability.

Second, MM reduces noise from the spurious affixes that the unsupervised segmenter induces for large lexicons. As Table 2 shows, the segmenter

Decoding	Level	Rec	Prec	F1	Acc
Gibbs	Type	82.9	83.2	83.1	74.5
	Token	87.5	89.1	88.3	86.7
MM	Type	95.9	95.8	95.9	93.9
	Token	97.3	94.0	95.6	95.1

Table 1: Comparison of agreement in outputs between 25 runs of Gibbs sampling vs. 2 runs of MM on the full MT-08 data set. We give the average segmentation recall, precision, F1-measure, and exact-match accuracy between outputs, at word-type and word-token levels.

	ATB	MT-08		
	GS	GS	MM	Morf
Unique prefixes	17	130	93	287
Unique suffixes	41	261	216	241
Top-95 prefixes	7	7	6	6
Top-95 suffixes	14	26	19	19

Table 2: Affix statistics of unsupervised segmenters. For the ATB lexicon, we show statistics for the Lee segmenter with regular Gibbs sampling (GS). For the MT-08 lexicon, we also show the output of the Lee segmenter with maximum marginal decoding (MM). In addition, we show statistics for Morfessor.

induces 130 prefixes and 261 suffixes for MT-08 (statistics for Morfessor are similar). This phenomenon is fundamental to Bayesian nonparametric models, which expand indefinitely to fit the data they are given (Wasserman, 2006). But MM helps to alleviate it, reducing unique prefixes and suffixes for MT-08 by 28% and 21%, respectively. It also reduces the number of unique prefixes/suffixes which account for 95% of the prefix/suffix tokens (*Top-95*).

Finally, we find that in our setting, MM increases accuracy not just over the mean, but over even the *best*-scoring of the runs. As shown in Table 3, MM increases segmentation F-measure from 86.2% to 88.2%. This exceeds the best published results on ATB (Naradowsky and Toutanova, 2011).

These results suggest that MM may be worth considering for other GS applications, not only for the accuracy improvements pointed out by Johnson and Goldwater (2009), but also for its potential to provide more stable and less noisy results.

Model	Mean	Min	Max	MM
M1	80.1	79.0	81.5	81.8
M2	81.4	80.2	83.0	82.0
M3	81.4	80.1	82.8	83.2
M4	86.2	85.4	87.2	88.2

Table 3: Segmentation F-scores on ATB dataset for Lee segmenter, shown for each Model level M1–M4 on the Arabic segmentation dataset used by (Poon et al., 2009): We give the mean, minimum, and maximum F-scores for 25 independent runs of Gibbs sampling, together with the F-score from running MM over that same set of runs.

5 MT Evaluation

5.1 Experimental Design

MT System. Our experiments were performed using a state-of-the-art, hierarchical string-to-dependency-tree MT system, described in Shen et al. (2008).

Morphological Analyzers. We compare the Lee segmenter with the supervised MSA segmenter MADA, using its "D3" scheme. We also compare with Sakhr, an intensively-engineered, supervised MSA segmenter which applies multiple NLP technologies to the segmentation problem, and which has given the best results for our MT system in previous work (Zbib et al., 2012a). We also compare with Morfessor.

MT experiments. We apply the appropriate segmenter to split words into morphemes, which we then treat as words for alignment and decoding. Following Lee et al. (2011), we segment the test and training sets jointly, estimating separate translation models for each segmenter/dataset combination.

Training and Test Corpora. Our "Full MSA" corpus is the NIST MT-08 Constrained Data Track Arabic training corpus (35M total, 336K unique words); our "Small MSA" corpus is a 1.3M-word subset. Both are tested on the MT-08 evaluation set. For dialect, we use a Levantine dialectal Arabic corpus collected from the web with 1.5M total, 160K unique words and 18K words held-out for test (Zbib et al., 2012b)

Performance Metrics. We evaluate MT with BLEU score. To calculate statistical significance, we use the boot-strap resampling method of Koehn (2004).

5.2 Results and Discussion

Table 4 summarizes the BLEU scores obtained from using various segmenters, for three training/test sets: Full MSA, Small MSA, and Levantine dialect.

As expected, Sakhr gives the best results for MSA. Morfessor underperforms the other segmenters, perhaps because of its lower accuracy on Arabic, as reported by Poon et al. (2009). The Lee segmenter gives the best results for Levantine, inducing valid Levantine affixes (e.g "hAl+" for MSA's "h*A-Al+", English "this-the") and yielding an 18% relative gain over the unsegmented baseline.

What is more surprising is that the Lee segmenter compares favorably with the supervised MSA segmenters on MSA itself. In particular, the Lee segmenter with MM yields higher BLEU scores than does MADA, a leading supervised segmenter, while preserving almost the same performance as GS on dialect. On Small MSA, it recoups 93% of even Sakhr's gain.

By contrast, the Lee segmenter recoups only 79% of Sakhr's gain on Full MSA. This might result from the phenomenon alluded to in Section 4, where additional data sometimes degrades performance for unsupervised analyzers. However, the Lee segmenter's gain on Levantine (18%) is higher than its gain on Small MSA (13%), even though Levantine has more data (1.5M vs. 1.3M words). This might be because dialect, being less standardized, has more orthographic and morphological variability, which unsupervised segmentation helps to resolve.

These experiments also show that while Model 4 gives the best F-score, Model 3 gives the best MT scores. Comparison of Model 3 and 4 segmentations shows that Model 4 induces a much larger number of inflectional suffixes, especially the feminine singular suffix "-p", which accounts for a plurality (16%) of the differences by token. While such suffixes improve F-measure on the segmentation references, they do not correspond to any English lexical unit, and thus do not help alignment.

An interesting question is how much performance might be gained from a supervised segmenter that was as intensively engineered for dialect as Sakhr was for MSA. Assuming a gain ratio of 0.93, similar to Small MSA, the estimated BLEU score would be 20.38, for a relative gain of just 5% over the unsupervised segmenter. Given the large engineering effort that would be required to achieve this gain, the unsupervised segmenter may be a more cost-effective choice for dialectal Arabic.

System		Small MSA	Full MSA	Lev Dial
Unsegmented		38.69	43.45	17.10
Sakhr		***43.99***	***46.51***	19.60
MADA		43.23	45.64	19.29
Morfessor		42.07	44.71	18.38
Lee GS	M1	43.12	44.80	19.70
	M2	43.16	45.45	***20.15+***
	M3	43.07	44.82	19.97
	M4	42.93	45.06	19.55
Lee MM	M1	43.53	45.14	19.75
	M2	43.45	45.29	19.75
	M3	**43.64+**	**45.84**	**20.09**
	M4	43.56	45.16	19.93

Table 4: BLEU scores for all experiments. Full MSA is the the full MT-08 corpus, Small MSA is a 1.3M-word subset, Lev Dial our Levantine dataset. For each of these, the highest Lee segmenter score is in bold, with "+" if statistically significant vs. MADA at the 95% confidence level or higher. The highest overall score is in bold italic.

6 Conclusion

We compare unsupervised vs. supervised morphological segmentation for Arabic-to-English machine translation. We add maximum marginal decoding to the unsupervised segmenter, and show that it surpasses the state-of-the-art segmentation performance, purges the segmenter of noise and variability, yields BLEU scores on MSA competitive with those from supervised segmenters, and gives an 18% relative BLEU gain on Levantine dialectal Arabic.

Acknowledgements
This material is based upon work supported by DARPA under Contract Nos. HR0011-12-C00014 and HR0011-12-C00015, and by ONR MURI Contract No. W911NF-10-1-0533. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the US government. We thank Rabih Zbib for his help with interpreting Levantine Arabic segmentation output.

References

Eleftherios Avramidis and Philipp Koehn. 2008. Enriching morphologically poor languages for statistical machine translation. In *Proceedings of ACL-08: HLT*.

Ibrahim Badr, Rabih Zbib, and James Glass. 2008. Segmentation for English-to-Arabic statistical machine translation. In *Proceedings of ACL-08: HLT, Short Papers*.

Ann Clifton and Anoop Sarkar. 2011. Combining morpheme-based machine translation with post-processing morpheme prediction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech Lang. Process.*, 4:3:1–3:34, February.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of ACL*.

Nizar Habash. 2008. Four techniques for online handling of out-of-vocabulary words in Arabic-English statistical machine translation. In *Proceedings of ACL-08: HLT, Short Papers*.

Mark Johnson and Sharon Goldwater. 2009. Improving nonparametric 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*.

Philipp Koehn and Hieu Hoang. 2007. Factored translation models. In *Proceedings of EMNLP-CoNLL*, pages 868–876.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*.

Yoong Keok Lee, Aria Haghighi, and Regina Barzilay. 2011. Modeling syntactic context improves morphological segmentation. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*.

Minh-Thang Luong, Preslav Nakov, and Min-Yen Kan. 2010. A hybrid morpheme-word representation for machine translation of morphologically rich languages. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Coşkun Mermer and Ahmet Afşın Akın. 2010. Unsupervised search for the optimal segmentation for statistical machine translation. In *Proceedings of the ACL 2010 Student Research Workshop*, pages 31–36, Uppsala, Sweden, July. Association for Computational Linguistics.

Coşkun Mermer and Murat Saraclar. 2011. Unsupervised Turkish morphological segmentation for statistical machine translation. In *Workshop on Machine Translation and Morphologically-rich languages*, January.

Preslav Nakov and Hwee Tou Ng. 2011. Translating from morphologically complex languages: A paraphrase-based approach. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Jason Naradowsky and Kristina Toutanova. 2011. Unsupervised bilingual morpheme segmentation and alignment with context-rich hidden semi-Markov models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

NIST. 2010. NIST 2008 Open Machine Translation (Open MT) Evaluation. `http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2010T21/`.

Kemal Oflazer. 1993. Two-level description of Turkish morphology. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Lawrence R. Rabiner. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286.

Fatiha Sadat and Nizar Habash. 2006. Combination of Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*.

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*.

Sami Virpioja, Jaakko J. Väyrynen, Mathias Creutz, and Markus Sadeniemi. 2007. Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. In *Proceedings of the Machine Translation Summit XI*.

Larry Wasserman. 2006. *All of Nonparametric Statistics*. Springer.

Mei Yang and Katrin Kirchhoff. 2006. Phrase-based backoff models for machine translation of highly inflected languages. In *Proceedings of EACL*.

Rabih Zbib, Michael Kayser, Spyros Matsoukas, John Makhoul, Hazem Nader, Hamdy Soliman, and Rami Safadi. 2012a. Methods for integrating rule-based and statistical systems for Arabic to English machine translation. *Machine Translation*, 26(1-2):67–83.

Rabih Zbib, Erika Malchiodi, Jacob Devlin, David Stallard, Spyros Matsoukas, Richard Schwartz, John Makhoul, Omar F. Zaidan, and Chris Callison-Burch. 2012b. Machine translation of Arabic dialects. In *NAACL 2012: Proceedings of the 2012 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Montreal, Quebec, Canada, June. Association for Computational Linguistics.

A Meta Learning Approach to Grammatical Error Correction

Hongsuck Seo[1], Jonghoon Lee[1], Seokhwan Kim[2], Kyusong Lee[1]
Sechun Kang[1], Gary Geunbae Lee[1]
[1]Pohang University of Science and Technology
[2]Institute for Infocomm Research
{hsseo, jh21983}@postech.ac.kr, kims@i2r.a-star.edu.sg
{kyusonglee, freshboy, gblee}@postech.ac.kr

Abstract

We introduce a novel method for grammatical error correction with a number of small corpora. To make the best use of several corpora with different characteristics, we employ a meta-learning with several base classifiers trained on different corpora. This research focuses on a grammatical error correction task for article errors. A series of experiments is presented to show the effectiveness of the proposed approach on two different grammatical error tagged corpora.

1. Introduction

As language learning has drawn significant attention in the community, grammatical error correction (GEC), consequently, has attracted a fair amount of attention. Several organizations have built diverse resources including grammatical error (GE) tagged corpora.

Although there are some publicly released GE tagged corpora, it is still challenging to train a good GEC model due to the lack of large GE tagged learner corpus. The available GE tagged corpora are mostly small datasets having different characteristics depending on the development methods, e.g. spoken corpus vs. written corpus. This situation forced researchers to utilize native corpora rather than GE tagged learner corpora for the GEC task.

The native corpus approach consists of learning a model that predicts the correct form of an article given the surrounding context. Some researchers focused on mining better features from the linguistic and pedagogic knowledge, whereas others focused on testing different classification methods (Knight and Chandler, 1994; Minnen et al., 2000; Lee, 2004; Nagata et al., 2006; Han et al., 2006; De Felice, 2008).

Recently, a group of researchers introduced methods utilizing a GE tagged learner corpus to derive more accurate results (Han et al., 2010; Rozovskaya and Roth, 2010). Since the two approaches are closely related to each other, they can be informative to each other. For example, Dahlmeier and Ng (2011) proposed a method that combines a native corpus and a GE tagged learner corpus and it outperformed models trained with either a native or GE tagged learner corpus alone. However, methods which train a GEC model from various GE tagged corpora have received less focus.

In this paper, we present a novel approach to the GEC task using meta-learning. We focus mainly on article errors for two reasons. First, articles are one of the most significant sources of GE for the learners with various L1 backgrounds. Second, the effective features for article error correction are already well engineered allowing for quick analysis of the method. Our approach is distinguished from others by integrating the predictive models trained on several GE tagged learner corpora, rather than just one GE tagged corpus. Moreover, the framework is compatible to any classification technique. In this study, we also use a native corpus employing Dahlmeier and Ng's approach. We demonstrate the effectiveness of the proposed method against baseline models in article error correction tasks.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 328–332,
Jeju, Republic of Korea, 8-14 July 2012.

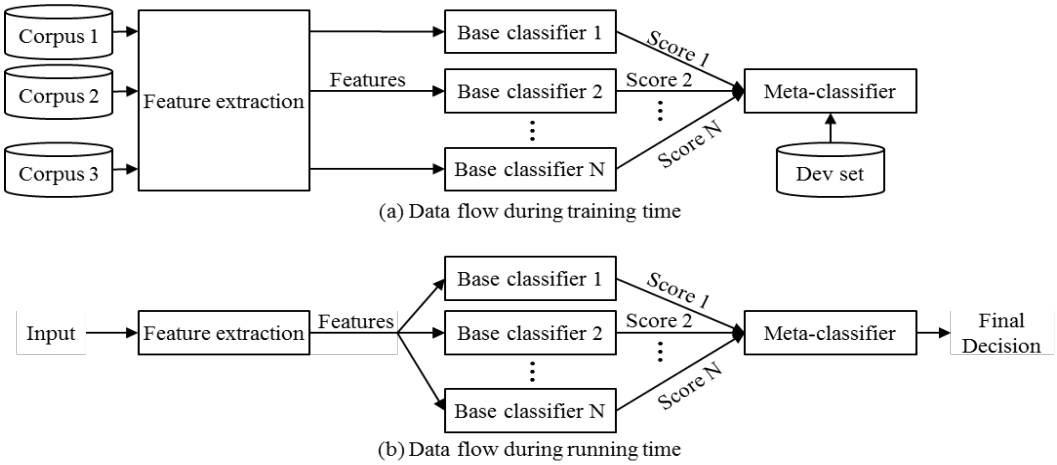


Figure 1: Overview of the proposed method

The remainder of this paper is organized as follows: Section 2 explains our proposed method. The experiments are presented in Section 3. Finally, Section 4 concludes the paper.

2. Method

Our method predicts the type of article for a noun phrase within three classes: *null*, *definite*, and *indefinite*. A correction arises when the prediction disagrees with the observed article. The meta-learning technique is applied to this task to deal with multiple corpora obtained from different sources.

A meta-classifier decides the final output based on the intermediate results obtained from several base classifiers. Each base classifier is trained on a different corpus than are the other classifiers. In this work, the feature extraction processes used for the base classifiers are identical to each other for simplicity, although they need not necessarily be identical. The meta-classifier takes the output scores of the base classifiers as its input and is trained on the held-out development data (Figure 1a). During run time, the trained classifiers are organized in the same manner. For the given features, the base classifiers independently calculate the score, then the meta-classifier makes the final decision based on the scores (Figure 1b).

2.1. Meta-learning

Meta-learning is a sequential learning process following the output of other base learners (classifiers). Normally, different classifiers successfully predict results on different parts of the input space, so researchers have often tried to combine different classifiers together (Breiman, 1996; Cohen et al., 2007; Zhang, 2007; Aydın, 2009; Menahem et al., 2009). To capitalize on the strengths and compensate for the weaknesses of each classifier, we build a meta-learner that takes an input vector consisting of the outputs of the base classifiers. The performance of meta-learning can be improved using output probabilities for every class label from the base classifiers.

The meta-classifier for the proposed method consists of multiple linear classifiers. Each classifier takes an input vector consisting of the output scores of each base classifier and calculates a score for each type of article. The meta-classifier finally takes the class having the maximum score.

A common design of an ensemble is to train different base classifiers with the same dataset, but in this work one classification technique was used with different datasets each having different characteristics. Although only one classification method was used in this work, different methods each well-tuned to the individual corpora may be used to improve the performance.

We employed the meta-learning method to generate synergy among corpora with diverse characteristics. More specifically, it is shown by cross validation that meta-learning performs at a level that is comparable to the best base classifier (Dzeroski and Zenko, 2004).

2.2. Base Classifiers

In the meta-learning framework, the performance of the base classifiers is important because the improvement in base classification generally enha-

nces the overall performance. The base classifiers can be expected to become more informative as more data are provided. We followed the structural learning approach (Ando and Zhang, 2005), which trains a model from both a native corpus and a GE tagged corpus (Dahlmeire and Ng, 2011), to improve the base classifiers by the additional information extracted from a native corpus.

Structural learning is a technique which trains multiple classifiers with common structure. The common structure chooses the hypothesis space of each individual classifier and the individual classifiers are trained separately once the hypothesis space is determined. The common structure can be obtained from auxiliary problems which are closely related to the main problems.

A word selection problem is a task to predict the appropriate word given the surrounding context in a native corpus and is a closely related auxiliary problem of the GEC task. We can obtain the common structure from the article selection problem and use it for the correction problem.

In this work, all the base classifiers used the same least squares loss function for structural learning. We adopted the feature set investigated in De Felice (2008) for article error correction. We use the Stanford coreNLP toolkit[1] (Toutanova and Manning, 2000; Klein and Manning, 2003a; Klein and Manning, 2003b; Finkel et al, 2005) to extract the features.

2.3. Evaluation Metric

The effectiveness of the proposed method is evaluated in terms of accuracy, precision, recall, and F_1-score (Dahlmeire and Ng, 2011). Accuracy is the number of correct predictions divided by the total number of instances. Precision is the ratio of the suggested corrections that agree with the tagged answer to the total number of the suggested corrections whereas recall is the ratio of the suggested corrections that agree with the tagged answer to the total number of corrections in the corpus.

3. Experiments

3.1. Datasets

In this work we used a native corpus and two GE tagged corpora. For the native corpus, we used news data[2] which is a large English text extracted from news articles. The First Certificate in English exams in the Cambridge Learner Corpus [3] (hereafter, CLC-FCE; Yannakoudakis et al., 2011) and the Japanese Learner English corpus (Izumi et. al., 2005) were used for the GE tagged corpora.

We extracted noun phrases from each corpus by parsing the text of the respective corpora. (1) We parsed the native corpus from the beginning until approximately a million noun phrases are extracted. (2) About 90k noun phrases containing ~3,300 mistakes in article usage were extracted from the entire CLC-FCE corpus, and (3) about 30k noun phrases containing ~2,500 mistakes were extracted from the JLE corpus.

The extracted noun phrases were used for our training and test data. We hold out 10% of the data for the test. We applied 20% under-sampling to the training instances that do not have any errors to alleviate data imbalance in the training set.

We emphasize the fact that the two learner corpora differ from each other in three aspects. The first aspect is the styles of the texts: the CLC is literary whereas the JLE is colloquial. The second is the error rate: about 3.5% for CLC-FCE and 8.5% for JLE. Finally, the third is the distribution of L1 languages of the learners: the learners of the CLC corpus have various L1 backgrounds whereas the learners of the JLE consist of only Japanese. These experiments demonstrate the effectiveness of the proposed method relying on the diversity of the corpora.

The native corpus was used to find the common structure using structural learning and two GE tagged learner corpora are used to train the base classifiers by structural learning with the common structure obtained from the news corpus.

We trained three classifiers for comparison; (1) the classifier (INTEG) trained with the integrated training set of the two GE tagged corpora, and two base classifiers used for the ensemble: (2) the base classifier (CB) trained only with the CLC-FCE and (3) the other base classifier (JB) trained with the JLE.

3.2. Results

The accuracy obtained from the word selection task with the news corpus was 76.10%. Upon

[1] http://nlp.stanford.edu/software/corenlp.shtml

[2] http://www.statmt.org/wmt09/translation-task.html

[3] http://www.ilexir.com/

Model	Acc.	Prec.	Rec.	F_1
INTEG	73.37	4.69	**72.39**	8.82
CB	77.20	5.39	71.17	10.03
Proposed	**86.99**	**6.17**	45.77	**10.88**

Table 1: Best results for GEC task on CLC-FCE test set.

Model	Acc.	Prec.	Rec.	F_1
INTEG	78.87	14.88	85.47	25.35
JB	78.02	14.49	**86.32**	24.82
Proposed	**89.61**	**19.28**	46.60	**27.27**

Table 2: Best results for GEC task on JLE test set.

Model	Acc.	Prec.	Rec.	F_1
INTEG	74.64	6.84	**77.86**	12.58
Proposed	**87.50**	**8.61**	46.12	**14.52**

Table 3: Best results for GEC task on the integrated set of CLC-FCE and JLE test sets.

obtaining the parameters of the word selection task, the structural parameter Θ was calculated by singular value decomposition and was used for the structural learning of the main GEC task.

We used three different test data sets: the CLC-FCE, the JLE and an integrated test set of the two. The accuracy (Acc.) and the precision (Prec.) of the INTEG was poorer than CB on the CLC-FCE test set (Table 1), whereas INTEG outperformed JB on the JLE test (Table 2).

Some instances extracted from the CLC-FCE corpus have similar characteristics to the instances from the JLE corpus. This overlap of instances affected the performance in both positive and negative ways. Prediction of instances similar to those in the JLE was enhanced. Consequently, INTEG model demonstrated better accuracy and precision for the JLE test set. Unfortunately, for the CLC test set, the instances resulted in lower accuracy and precision.

The proposed model is able to alleviate this model bias due to similar instances observed in the INTEG model. The accuracy of the proposed model consistently increased by over 10% for all three data sets. The relative performance gain in terms of F1-score (F_1) was 15% on the integrated set. This performance gain stems from the over 25% relative improvement of the precision (Table 1, 2 and 3).

We believe the improvement comes from the contribution of reconfirming procedures performed by the meta-classifier. When the prediction of the two base classifiers conflicts with each other, the meta-classifier tends to choose the one with a higher confidence score; this choice improves the accuracy and precision because known features generate a higher confidence whereas unseen or less-weighted features generate a lower score.

Although the proposed model introduced a tradeoff between precision and recall (Rec.), this tradeoff was tolerable in order to improve the overall F1-score. Since GEC is a task where false alarm is critical, obtaining high precision is very important. The low precision on the whole experiments is due to the data imbalance. Instances in the dataset are mostly not erroneous, e.g., only 3.5% of erroneous instances for the CLC corpus. The standard for correct prediction is also very strict and does not allow multiple answers. Performance can be evaluated in a more realistic way by applying a softer standard, e.g., by evaluating manually.

4. Conclusion

We have presented a novel approach to grammatical error correction by building a meta-classifier using multiple GE tagged corpora with different characteristics in various aspects. The experiments showed that building a meta-classifier overcomes the interference that occurs when training with a set of heterogeneous corpora. The proposed method also outperforms the base classifier themselves tested on the same class of test set as the training set with which the base classifiers are trained. A better automatic evaluation metric would be needed as further research.

Acknowledgments

Industrial Strategic technology development program, 10035252, development of dialog-based spontaneous speech interface technology on mobile platform, funded by the Ministry of Knowledge Economy (MKE, Korea).

References

R.K. Ando and T. Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6, pp. 1817-1853.

U. Aydın, S. Murat, Olcay T Yıldız, A. Ethem, 2009, Incremental construction of classifier and discriminant ensembles, *Information Science*, 179 (9), pp. 144-152.

L. Breiman, 1996, Bagging predictors, *Machine Learning*, pp. 123–140.

S. Cohen, L. Rokach, O. Maimon, 2007, Decision tree instance space decomposition with grouped gain-ratio, *Information Science*, 177 (17), pp. 3592–3612.

D. Dahlmeier, H. T. Ng, 2011, Grammatical error correction with alternating structure optimization, In *Proceedings of the 49th Annual Meeting of the ACL-HLT 2011,* pp. 915-923.

R. De Felice. 2008. *Automatic Error Detection in Non-native English*. Ph.D. thesis, University of Oxford.

S. Dzeroski, B. Zenko, 2004, Is combining classifiers with stacking better than selecting the best one?, *Machine Learning*, 54 (3), pp. 255–273.

J. R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43nd Annual Meeting of the ACL*, pp. 363-370.

N.R. Han, M. Chodorow, and C. Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(02), pp. 115-129.

N.R. Han, J. Tetreault, S.H. Lee, and J.Y. Ha. 2010. Using an error-annotated learner corpus to develop an ESL/EFL error correction system. In *Proceedings of LREC*.

D. Klein and C.D. Manning. 2003a. Accurate unlexicalized parsing. In *Proceedings of ACL*, pp. 423-430.

D. Klein and C.D. Manning. 2003b. Fast exact inference with a factored model for natural language processing. *Advances in Neural Information Processing Systems (NIPS 2002)*, 15, pp. 3-10.

K. Knight and I. Chander. 1994. Automated postediting of documents. In *Proceedings of AAAI*, pp. 779-784.

J. Lee. 2004. Automatic article restoration. In *Proceedings of HLT-NAACL*, pp. 31-36.

R. Nagata, A. Kawai, K. Morihiro, and N. Isu. 2006. A feedback-augmented method for detecting errors in the writing of learners of English. In *Proceedings of COLING-ACL*, pp. 241--248.

A. Mariko, 2007, Grammatical errors across proficiency levels in L2 spoken and written English, *The Economic Journal of Takasaki City University of Economics*, 49 (3, 4), pp. 117-129.

E. Menahem, L. Rokach, Y. Elovici, 2009, Troika-An imporoved stacking schema for classification tasks, *Information Science*, 179 (24), pp. 4097-4122.

G. Minnen, F. Bond, and A. Copestake. 2000. Memory-based learning for article generation. In *Proceedings of CoNLL*, pp. 43-48.

E. Izumi, K. Uchimoto, H. Isahara, 2005, Error annotation for corpus of Japanese learner English, In *Proceedings of the 6th International Workshop on Linguistically Interpreted Corpora*, pp. 71-80.

A. Rozovskaya and D. Roth. 2010. Training paradigms for correcting errors in grammar and usage. In *Proceedings of HLT-NAACL*, pp. 154-162.

K. Toutanova and C. D. Manning. 2000. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. *In Proceedings of the Joint SIGDAT Conference on EMNLP/VLC-2000*, pp. 63-70.

H.Yannakoudakis, T. Briscoe, B. Medlock, 2011, A new dataset and method for automatically grading ESOL texts, In *Proceedings of ACL*, pp. 180-189.

G. P. Zhang, 2007, A neural network ensemble method with jittered training data for time series forecasting, *Information Sciences: An International Journal*, 177 (23), pp. 5329–5346.

Fine Granular Aspect Analysis using Latent Structural Models

Lei Fang[1] and **Minlie Huang**[2]
State Key Laboratory of Intelligent Technology and Systems,
Tsinghua National Laboratory for Information Science and Technology,
Department of Computer Science and Technology,
Tsinghua University, Beijing 100084, PR China.
[1]fang-l10@mails.tsinghua.edu.cn
[2]aihuang@tsinghua.edu.cn

Abstract

In this paper, we present a structural learning model for joint sentiment classification and aspect analysis of text at various levels of granularity. Our model aims to identify highly informative sentences that are aspect-specific in online custom reviews. The primary advantages of our model are two-fold: first, it performs document-level and sentence-level sentiment polarity classification jointly; second, it is able to find informative sentences that are closely related to some respects in a review, which may be helpful for aspect-level sentiment analysis such as aspect-oriented summarization. The proposed method was evaluated with 9,000 Chinese restaurant reviews. Preliminary experiments demonstrate that our model obtains promising performance.

1 Introduction

Online reviews have been a major resource from which users may find opinions or comments on the products or services they want to consume. However, users sometimes might be overwhelmed, and not be able to read reviews one by one when facing a considerably large number of reviews, and they may be not be satisfied by only being served with document-level reviews statistics (that is, the number of reviews with 1-star, 2-star, ..., respectively). Aspect-level review analysis may be alternative for addressing this issue as aspect-specific opinions may more clearly, explicitly, and completely describe the quality of a product from different properties.

Our goal is to discover informative sentences that are consistent with the overall rating of a review, and simultaneously, to perform sentiment analysis at aspect level. Notice, that a review with a high rating (say, 4/5 stars) may contain both negative and positive opinions, and the same to a review with a very low rating (say, 1/2 star). From our point of view, each review has a set of sentences that are informative and coherent to its overall rating. To perform fine granular sentiment analysis, the first step is to discover such coherent content.

Many information needs require the systems to perform fine granular sentiment analysis. Aspect-level sentiment analysis may be more useful for users to have a global picture of opinions on the product's properties. Furthermore, different users may have different preferences on different aspects of a product. Taking the reviews on mobile phones as an example, female users may focus more on the appearance while male users may lay more emphasis on the hardware configuration; younger users prefer to the app or game resources while older users may just pay attention to the basic function of calling or messaging.

In recent years, there has been much work focused multilevel sentiment classification using structural learning models. Yi (2007) extends the standard conditional random fields to model the local sentiment flow. Ryan (2007) proposed structured models for fine-to-coarse sentiment analysis. Oscar (2011) proposed to discover fine-grained sentiment with hidden-state CRF(Quattoni et al., 2007). Yessenalina (2010) deployed the framework of latent structural SVMs(Yu and Joachims., 2009) for multilevel sentiment classification. As for aspect level rating, ranking, or summarization, Benjamin(2007) em-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 333–337,
Jeju, Republic of Korea, 8-14 July 2012.

ployed the good grief algorithm for multiple aspect ranking and the extensions of the generative topic models were also widely studied, such as (Titov and McDonald., 2008; Brody and Elhadad., 2010; Wang et al., 2010; Li et al., 2011; Lu et al., 2011; Jo and Oh., 2011; Lin and He, 2009).

In this paper, we build a general structural learning model for joint sentiment classification and aspect analysis using a latent discriminate method. Our model is able to predict the sentiment polarity of document as well as to identify aspect-specific sentences and predict the polarity of such sentences. The proposed method was evaluated with 9,000 Chinese restaurant reviews. Preliminary experiments demonstrate that our model obtains promising performance.

2 Model

2.1 Document Structure

We assume that the polarity of document is closely related to some aspects for the reason that people are writing reviews to praise or criticize certain aspects. Therefore, each informative sentence of the document characterizes one aspect, expressing aspect specific polarity or subjective features. Similar to previous work on aspect analysis (Wang et al., 2010) and multi-level sentiment classification (Yessenalina et al., 2010), we define the aspect as a collection of synonyms. For instance, the word set {"value", "price", "cost", "worth", "quality"} is a synonym set corresponding to the aspect "price". For each document, an aspect is described by one or several sentences expressing aspect specific polarity or subjective information.

Let document be denoted by x, and $y \in \{+1, -1\}$ represents the positive or negative polarity of the document, s is the set of informative sentences, in which each sentence is attached with certain aspect $a_i \in A = \{a_1, ..., a_k\}$. Yessenalina (2010) chooses a sentence set that best explains the sentiment of the whole document while the s here retain this property. Let $\Psi(x, y, s)$ denote the joint feature map that outputs the features describing the quality of predicting sentiment y using the sentence set s.

Let x^j denote the j-th sentence of document x, and a^j is the attached aspect of x^j. In spirit to (Yessenalina et al., 2010), we propose the following formulation of the discriminate function

$$\vec{w}^T \Psi(x, y, s) = \frac{1}{N(x)} \sum_{j \in s} \left(y \cdot \vec{w}^T_{pol_{a^j}} \psi_{pol}(x^j) + \vec{w}^T_{subj_{a^j}} \psi_{subj}(x^j) \right)$$

where $N(x)$ is the normalizing factor, $\psi_{pol}(x^j)$ and $\psi_{subj}(x^j)$ represents the polarity and subjectivity features of sentence x^j respectively. $\vec{w}_{pol}$ and $\vec{w}_{subj}$ denote the weight for polarity and subjectivity features. To be specific for each aspect, we have $\vec{w}_{pol_a}$ and $\vec{w}_{subj_a}$ representing the vector of feature weight for aspect a to calculate the polarity and subjectivity score.

$$\vec{w}^T_{pol} = \begin{bmatrix} \vec{w}^T_{pol_{a_0}} \\ \vdots \\ \vec{w}^T_{pol_{a_k}} \end{bmatrix}, \quad \vec{w}^T_{subj} = \begin{bmatrix} \vec{w}^T_{subj_{a_0}} \\ \vdots \\ \vec{w}^T_{subj_{a_k}} \end{bmatrix}$$

To make prediction, we have the document-level sentiment classifier as

$$h(x; \vec{w}) = \underset{y=\pm 1}{\operatorname{argmax}} \max_{s \in S(x)} \vec{w}^T \Psi(x, y, s)$$

where $S(x) = \{s \subseteq 1, \ldots, |x| : |s| \leq f(|x|)\}$, $f(|x|)$ is a function that depends only on the number of sentences in x, as illustrated in (Yessenalina et al., 2010). Therefore, for each sentence x^j, we compute the joint subjectivity and polarity score with respect to aspect a and label y as

$$score(x^j, a, y) = y \cdot \vec{w}^T_{pol_a} \psi_{pol}(x^j) + \vec{w}^T_{subj_a} \psi_{subj}(x^j)$$

we then assign aspect a^j to sentence x^j if

$$a^j = \underset{a \in A}{\operatorname{argmax}} score(x^j, a, y)$$

After sorting $score(x^j, a^j, y)$ in decreasing order and taking summation by selecting the top $f(|x|)$ (or fewer, if there are fewer than $f(|x|)$ that have positive joint score) sentences as the total score for each $y \in \{+1, -1\}$, we then predict y with the higher joint score as the sentiment of the whole document. This formulation of $\vec{w}^T \Psi(x, y, s)$ and classifier explains that for each sentence, the assigned aspect has the highest score over other aspects.

2.2 Feature Space

In our model, we use **bag-of-words** features. In order to obtain a model that is jointly trained, and satisfy the condition that the overall polarity of document should influence the sentiment of extracted informative sentences. We denote the weight vector modeling the polarity of entire document as $\vec{w}_{doc}$, as follows:

$$\vec{w}^T\Psi(x,y,s) = \frac{y}{N(x)}\left(\sum_{j\in s}(\vec{w}^T_{pol_{aj}}\psi_{pol}(x^j)+\vec{w}^T_{doc}\psi_{pol}(x^j))\right) + \frac{1}{N(x)}\left(\sum_{j\in s}\vec{w}^T_{subj_{aj}}\psi_{subj}(x^j)\right)+y\cdot\vec{w}^T_{doc}\psi_{pol}(x)$$

2.3 Training

We trained our model using the latent structural SVMs (Yu and Joachims., 2009).

OP1:

$$\min_{\vec{w},\xi\geq 0}\frac{1}{2}||w||^2+\frac{C}{N}\sum_{i=1}^{N}\xi_i$$

$$s.t.\forall i:$$

$$\max_{s\in S(x_i)}\vec{w}^T\Psi(x_i,y_i,s)\geq \max_{s'\in S(x_i)}\vec{w}^T\Psi(x_i,-y_i,s')+\triangle(y_i,-y_i,s')-\xi_i$$

We define $\triangle(y_i,-y_i,s')=1$, that is, we view document level sentiment classification loss as the loss function. It should be noticed that OP1 is non-convex. To circumvent the optimization difficulty, we employ the framework of structural SVMs (Tsochantaridis et al., 2004) with latent variables proposed by Yu (2009) using the CCCP algorithm (Yuille and Rangarajan., 2003). In terms of the formulation here, since the true informative sentence set is never observed, it is a hidden or latent variable. Thus, we keep s_i fixed to compute the upper bound for the concave part of each constraint, and rewrite the constraints as

$$\xi_i\geq\max_{s'\in S(x_i)}\vec{w}^T\Psi(x_i,-y_i,s')-\vec{w}^T\Psi(x_i,y_i,s_i)+1$$

After that, we have y_i completed with the latent variable s_i as if it is observed. For each training example, starting with an initialization sentence set in which each sentence is with an aspect label, the training procedure alternates between solving an instance of the structural SVM using the s_i and predicting a new sentence until the learned weight vector $\vec{w}$ converges. In our work, we use the performance on a validation set to choose the halting iteration, as is similar to Yessenalina (2010).

2.4 Model Initialization

To initialize the informative sentence set, following the experiment result of Yessenalina (2010), we set $f(|x|)=0.3*|x|$, that is, we only select the top 30% of the total sentences as the set of informative part of the document. The normalizing factor is set as $N(x)=\sqrt{f|x|}$, as Yessenalina (2010) demonstrates that square root normalization can be useful. To analyze the aspect of each sentence, we need to give an initial guess of the aspect and sentiment for each sentence.

Sentence level sentiment initialization : To initialize the sentence level sentiment, we employ a rule based method incorporating positive and negative sentiment terms, with adversative relation considered.

Sentence aspect assignment initialization : Obviously, if a synonym of aspect a occurs in sentence x^l, we assign aspect a to x^l, and add x^l to an aspect specific sentence set P_a.For sentence x^l without any aspect term, we set a as the aspect label if

$$a=\underset{a'\in A}{\text{argmax}}\, similarity(x^l,P_{a'})$$

We select the sentences whose sentiment is consistent with the overall rating of a review as the initial guess of the latent variable.

3 Experiments

In this section, we evaluate our model in terms of document and sentence level sentiment classification, we also analyze the performance of aspect assignment for each sentence. The model is evaluated on the Chinese restaurant reviews crawled from Dianping[1]. Each of the reviews has an overall rating ranging from one to five stars. To be specific, we consider a review as positive if its rating is greater

[1] http://www.dianping.com/

than or equal to 4 stars, or negative if less than or equal to 2 stars. The corpus has 4500 positive and 4500 negative reviews. Data and an implementation of our model are publicly available[2].

We train 5 different models by splitting these reviews into 9 folds. Two folds are left out as the testing set, and each model takes 5 folds as training set, 2 folds as validation set, and the performance is averaged. Besides, we also manually label 100 reviews, in which each sentence is labeled as positive or negative corresponding to certain aspect or with no aspect description. On average, each review has 9.66 sentences. However, only 21.5% of the total sentences can be assigned to aspect by directly matching with aspect terms, which explains that keywords based aspect sentiment analysis may fail. For restaurant reviews, we pre-defined 11 aspects, and for each aspect, we select about 5 frequently used terms to describe that aspect. Table 1 shows some examples of the aspect synonym set used in this paper:

Aspect	Synonym Set
Taste	味道"taste", 口味"flavor"
Price	价格"price" , 价钱"cost"
Dishes	菜品"dishes", 菜式"cuisine"
Ingredients	食物"food" , 食材"ingredients"
Facility	设施"facility", 座位"seat"
Location	位置"location",
Environment	环境"environment",
	装修"decoration"
Service	服务"service" , 服务员"waiter"
	态度"attitude"

Table 1: Samples of Aspect Synonym.

Document level sentiment classification We compare our method with previous work on sentiment classification using standard SVM(Pang et al., 2002). Our model yields an accuracy of 94.15% while the standard SVM classifier yields an accuracy of 90.35%. Clearly, our model outperforms the baseline on document level sentiment classification.

Sentence level sentiment classification Our method can extract a set of informative sentence that are coherent to the overall rating of a review. The evaluation of sentence-level sentiment classification is based on manual annotation. We sample 100 reviews, and present the extracted 300 sentences to annotators who have been asked to assign positive/negative/non-related labels. Among the sentences, 251 correctly classified as positive or negative while 49 are misclassified. And, 38 sentences of the 49 sentences have mix opinions or are non-subjective sentences.

Aspect Assignment To evaluate the accuracy of aspect assignment, we compare the predicted aspect labels with the ground truth (manual annotation). As some of sentences have explicit aspect terms and can be easily identified, we only consider those sentences without aspect words. In the extracted 300 sentences, 78 sentences have aspect terms, and for the rest, our model assigns correct aspect labels to 44 sentences while random guess only maps 21 sentences with right labels.

4 Conclusion and Future Work

In this paper, we address the task of multilevel sentiment classification of online custom reviews for fine granular aspect analysis. We present a structural learning model based on struct-SVM with latent variables. The informative sentence set is regarded as latent variable, in which each sentence is attached with certain aspect label. The training procedure alternates between solving an instance of the standard structural SVM optimization and predicting a new sentence set until the halting condition is satisfied. In addition, our model is a enough general model which can be easily extended to other domains. Preliminary experiments demonstrate that our model obtains promising performance.

There are several possibilities to improve our model. For future work, we propose to incorporate prior knowledge of latent variables to the model. One possible way is to reformulate the loss function by taking the predicted aspect of the extracted sentences into consideration. Another is to introduce confidence score to the extracted sentences, such that the learned support vectors that are labeled with higher confidence shall assert more force on the decision plane.

Acknowledgments

This paper was supported by Chinese 973 project under No.2012CB316301 and National Chinese Sci-

[2]http://www.qanswers.net/faculty/hml/

ence Foundation projects with No.60803075 and No.60973104.

References

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Proceedings of Annual Conference of the North American Chapter of the ACL, (NAACL)*.

Yohan Jo and Alice Oh. 2011. Aspect and sentiment unification model for online review analysis. In *Proceedings of Conference on Web Search and Data Mining (WSDM)*.

Peng Li, Yinglin Wang, Wei Gao, and Jing Jiang. 2011. Generating aspect-oriented multi-document summarization with event-aspect model. In *Proceedings of Conference on Empirical Methods in Natural Language Processing, (EMNLP)*.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the conference on Information and knowledge management(CIKM)*.

Bin Lu, Myle Ott, Claire Cardie, and Benjamin Tsou. 2011. Multi-aspect sentiment analysis with topic models. In *The ICDM' 2011 Workshop on Sentiment Elicitation from Natural Text for Information Retrieval and Extraction*.

Yi Mao and Guy Lebanon. 2007. Isotonic conditional random fields and local sentiment flow. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*.

Ryan McDonald, Kerry Hannan, Tyler Neylon, Mike Wells, and Jeff Reynar. 2007. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of Annual Meeting of the Association for Computational Linguistics, (ACL)*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

A. Quattoni, S. Wang, L.-P Morency, M. Collins, and T. Darrell. 2007. Hidden-state conditional random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Benjamin Snyder and Regina Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *Proceedings of Annual Conference of the North American Chapter of the ACL, (NAACL)*.

Oscar Täckström and Ryan McDonald. 2011. Discovering fine-grained sentiment with latent variable structured prediction models. In *Proceedings of Annual European Conference on Information Retrieval , (ECIR)*.

Ivan Titov and Ryan McDonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of Annual Meeting of the Association for Computational Linguistics, (ACL)*.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the International Conference on Machine Learning, (ICML)*.

Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: A rating regression approach. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*.

Ainur Yessenalina, Yisong Yue, and Claire Cardie. 2010. Multi-level structured models for document-level sentiment classification. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chun-Nam John Yu and Thorsten Joachims. 2009. Learning structural svms with latent variables. In *Proceedings of the International Conference on Machine Learning, (ICML)*.

A. L. Yuille and Anand Rangarajan. 2003. The concave-convex procedure (cccp). *Neural Computation*, 15:915–936.

Identifying High-Impact Sub-Structures for Convolution Kernels in Document-level Sentiment Classification

Zhaopeng Tu[†] **Yifan He**[‡§] **Jennifer Foster**[§] **Josef van Genabith**[§] **Qun Liu**[†] **Shouxun Lin**[†]

[†]Key Lab. of Intelligent Info. Processing [‡]Computer Science Department [§]School of Computing
Institute of Computing Technology, CAS New York University Dublin City University
[†]{tuzhaopeng,liuqun,sxlin}@ict.ac.cn,
[‡]yhe@cs.nyu.edu, [§]{jfoster,josef}@computing.dcu.ie

Abstract

Convolution kernels support the modeling of complex syntactic information in machine-learning tasks. However, such models are highly sensitive to the type and size of syntactic structure used. It is therefore an important challenge to automatically identify high impact sub-structures relevant to a given task. In this paper we present a systematic study investigating (combinations of) sequence and convolution kernels using different types of sub-structures in document-level sentiment classification. We show that minimal sub-structures extracted from constituency and dependency trees guided by a polarity lexicon show 1.45 point absolute improvement in accuracy over a bag-of-words classifier on a widely used sentiment corpus.

1 Introduction

An important subtask in sentiment analysis is sentiment classification. Sentiment classification involves the identification of positive and negative opinions from a text segment at various levels of granularity including *document-level*, *paragraph-level*, *sentence-level* and *phrase-level*. This paper focuses on document-level sentiment classification.

There has been a substantial amount of work on document-level sentiment classification. In early pioneering work, Pang and Lee (2004) use a flat feature vector (e.g., a bag-of-words) to represent the documents. A bag-of-words approach, however, cannot capture important information obtained from structural linguistic analysis of the documents. More recently, there have been several approaches which employ features based on deep linguistic analysis with encouraging results including Joshi and Penstein-Rose (2009) and Liu and Seneff (2009). However, as they select features manually, these methods would require additional labor when ported to other languages and domains.

In this paper, we study and evaluate diverse linguistic structures encoded as convolution kernels for the document-level sentiment classification problem, in order to utilize syntactic structures without defining explicit linguistic rules. While the application of kernel methods could seem intuitive for many tasks, it is non-trivial to apply convolution kernels to document-level sentiment classification: previous work has already shown that categorically using the entire syntactic structure of a single sentence would produce too many features for a convolution kernel (Zhang et al., 2006; Moschitti et al., 2008). We expect the situation to be worse for our task as we work with documents that tend to comprise dozens of sentences.

It is therefore necessary to choose appropriate substructures of a sentence as opposed to using the whole structure in order to effectively use convolution kernels in our task. It has been observed that not every part of a document is equally informative for identifying the polarity of the whole document (Yu and Hatzivassiloglou, 2003; Pang and Lee, 2004; Koppel and Schler, 2005; Ferguson et al., 2009): a film review often uses lengthy objective paragraphs to simply describe the plot. Such objective portions do not contain the author's opinion and are irrelevant with respect to the sentiment classifi-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 338–343,
Jeju, Republic of Korea, 8-14 July 2012.

cation task. Indeed, separating objective sentences from subjective sentences in a document produces encouraging results (Yu and Hatzivassiloglou, 2003; Pang and Lee, 2004; Koppel and Schler, 2005; Ferguson et al., 2009). Our research is inspired by these observations. Unlike in the previous work, however, we focus on syntactic *substructures* (rather than entire paragraphs or sentences) that contain subjective words.

More specifically, we use the terms in the lexicon constructed from (Wilson et al., 2005) as the indicators to identify the substructures for the convolution kernels, and extract different sub-structures according to these indicators for various types of parse trees (Section 3). An empirical evaluation on a widely used sentiment corpus shows an improvement of 1.45 point in accuracy over the baseline resulting from a combination of bag-of-words and high-impact parse features (Section 4).

2 Related Work

Our research builds on previous work in the field of sentiment classification and convolution kernels. For sentiment classification, the design of lexical and syntactic features is an important first step. Several approaches propose feature-based learning algorithms for this problem. Pang and Lee (2004) and Dave et al. (2003) represent a document as a bag-of-words; Matsumoto et al., (2005) extract frequently occurring connected subtrees from dependency parsing; Joshi and Penstein-Rose (2009) use a transformation of dependency relation triples; Liu and Seneff (2009) extract adverb-adjective-noun relations from dependency parser output.

Previous research has convincingly demonstrated a kernel's ability to generate large feature sets, which is useful to quickly model new and not well understood linguistic phenomena in machine learning, and has led to improvements in various NLP tasks, including relation extraction (Bunescu and Mooney, 2005a; Bunescu and Mooney, 2005b; Zhang et al., 2006; Nguyen et al., 2009), question answering (Moschitti and Quarteroni, 2008), semantic role labeling (Moschitti et al., 2008).

Convolution kernels have been used before in sentiment analysis: Wiegand and Klakow (2010) use convolution kernels for opinion holder extraction, Johansson and Moschitti (2010) for opinion expression detection and Agarwal et al. (2011) for sentiment analysis of Twitter data. Wiegand and Klakow (2010) use e.g. noun phrases as possible candidate opinion holders, in our work we extract any minimal syntactic context containing a subjective word. Johansson and Moschitti (2010) and Agarwal et al. (2011) process sentences and tweets respectively. However, as these are considerably shorter than documents, their feature space is less complex, and pruning is not as pertinent.

3 Kernels for Sentiment Classification

3.1 Linguistic Representations

We explore both sequence and convolution kernels to exploit information on surface and syntactic levels. For sequence kernels, we make use of lexical words with some syntactic information in the form of part-of-speech (POS) tags. More specifically, we define three types of sequences:

- SW, a sequence of lexical words, e.g.: *A tragic waste of talent and incredible visual effects*.
- SP, a sequence of POS tags, e.g.: *DT JJ NN IN NN CC JJ JJ NNS*.
- SWP, a sequence of words and POS tags, e.g.: *A/DT tragic/JJ waste/NN of/IN talent/NN and/CC incredible/JJ visual/JJ effects/NNS*.

In addition, we experiment with constituency tree kernels (CON), and dependency tree kernels (D), which capture hierarchical constituency structure and labeled dependency relations between words, respectively. For dependency kernels, we test with word (DW), POS (DP), and combined word-and-POS settings (DWP), and similarly for simple sequence kernels (SW, SP and SWP). We also use a vector kernel (VK) in a bag-of-words baseline. Figure 1 shows the constituent and dependency structure for the above sentence.

3.2 Settings

As kernel-based algorithms inherently explore the whole feature space to weight the features, it is important to choose appropriate substructures to remove unnecessary features as much as possible.

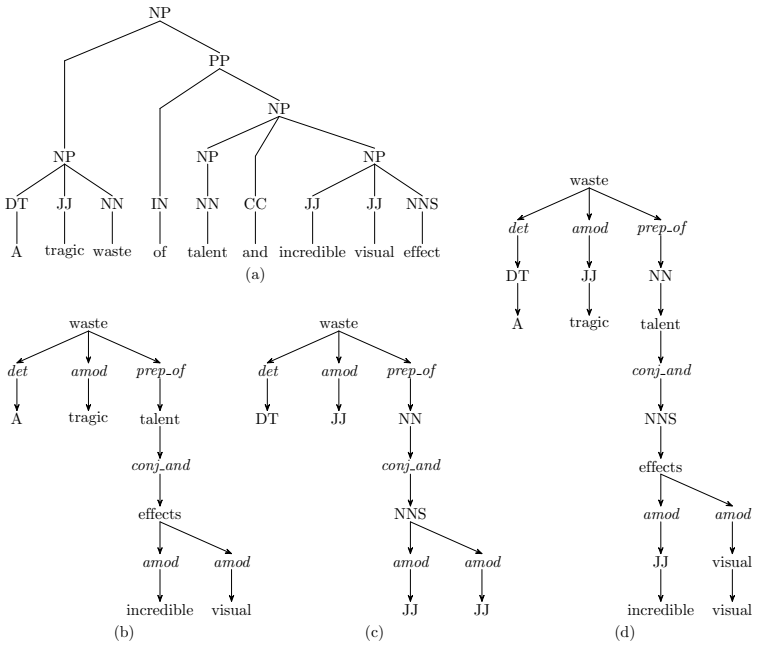


Figure 1: Illustration of the different tree structures employed for convolution kernels. (a) Constituent parse tree (CON); (b) Dependency tree-based words integrated with grammatical relations (DW); (c) Dependency tree in (b) with words substituted by POS tags (DP); (d) Dependency tree in (b) with POS tags inserted before words (DWP).

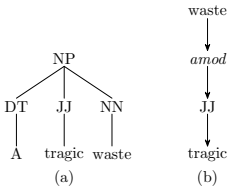


Figure 2: Illustration of the different settings on constituency (CON) and dependency (DWP) parse trees with *tragic* as the indicator word.

Scopes	Trees	Size
Document	32	24
Subjective Sentences	22	27
Constituent Substructures	30	10
Dependency Substructures	40	3

Table 1: The detail of the corpus. Here *Trees* denotes the average number of trees, and *Size* denotes the averaged number of words in each tree.

Unfortunately, in our task there exist several cues indicating the polarity of the document, which are distributed in different sentences. To solve this problem, we define the indicators in this task as subjective words in a polarity lexicon (Wilson et al., 2005). For each polarity indicator, we define the "scope" (the minimal syntactic structure containing at least one subjective word) of each indicator for different representations as follows:

For a constituent tree, a node and its children correspond to a grammatical production. Therefore, considering the terminal node *tragic* in the constituent structure tree in Figure 1(a), we extract the subtree rooted at the grandparent of the terminal, see Figure 2(a). We also use the corresponding sequence of words in the subtree for the sequential kernel.

For a dependency tree, we only consider the subtree containing the lexical items that are directly connected to the subjective word. For instance, given the node *tragic* in Figure 1(d), we will extract its direct parent *waste* integrated with dependency relations and (possibly) POS, as in Figure 2(b).

We further add two *background scopes*, one being subjective sentences (the sentences that contain subjective words), and the entire document.

4 Experiments

4.1 Setup

We carried out experiments on the movie review dataset (Pang and Lee, 2004), which consists of

1000 positive reviews and 1000 negative reviews. To obtain constituency trees, we parsed the document using the Stanford Parser (Klein and Manning, 2003). To obtain dependency trees, we passed the Stanford constituency trees through the Stanford constituency-to-dependency converter (de Marneffe and Manning, 2008).

We exploited Subset Tree (SST) (Collins and Duffy, 2001) and Partial Tree (PT) kernels (Moschitti, 2006) for constituent and dependency parse trees[1], respectively. A sequential kernel is applied for lexical sequences. Kernels were combined using plain (unweighted) summation. Corpus statistics are provided in Table 1.

We use a manually constructed polarity lexicon (Wilson et al., 2005), in which each entry is annotated with its degree of subjectivity (strong, weak), as well as its sentiment polarity (positive, negative and neutral). We only take into account the subjective terms with the degree of strong subjectivity.

We consider two baselines:

- **VK**: bag-of-words features using a *vector kernel* (Pang and Lee, 2004; Ng et al., 2006)
- **Rand**: a number of *randomly selected substructures* similar to the number of extracted substructures defined in Section 3.2

All experiments were carried out using the SVM-Light-TK toolkit[2] with default parameter settings. All results reported are based on 10-fold cross validation.

4.2 Results and Discussions

Table 2 lists the results of the different kernel type combinations. The best performance is obtained by combining VK and DW kernels, gaining a significant improvement of 1.45 point in accuracy. As far as PT kernels are concerned, we find dependency trees with simple words (DW) outperform both dependency trees with POS (DP) and those with both words and POS (DWP). We conjecture that in this case, as syntactic information is already captured by the dependency representation, POS tags can introduce little new information, and will add unnecessary complexity. For example, given the substructure *(waste (amod (JJ (tragic))))*, the PT kernel will use both *(waste (amod (JJ)))* and *(waste (amod (JJ (tragic))))*. We can see that the former is adding no value to the model, as the JJ tag could indicate either positive words (e.g. *good*) or negative words (e.g. *tragic*). In contrast, words are good indicators for sentiment polarity.

Kernels	Doc	Sent	Rand	Sub
VK	87.05			
VK + SW	87.25	86.95	87.25	87.40
VK + SP	87.35	86.95	87.45	87.35
VK + SWP	87.30	87.45	87.30	88.15*
VK + CON	87.45	87.65	87.45	88.30**
VK + DW	87.35	87.50	87.30	**88.50****
VK + DP	87.75*	87.20	87.35	87.75
VK + DWP	87.70*	87.30	87.65	87.80*

Table 2: Results of kernels. Here *Doc* denotes the whole document of the text, *Sent* denotes the sentences that contains subjective terms in the lexicon, *Rand* denotes randomly selected substructures, and *Sub* denotes the substructures defined in Section 3.2. We use "*" and "**" to denote a result is better than baseline VK significantly at $p < 0.05$ and $p < 0.01$ (sign test), respectively.

The results in Table 2 confirm two of our hypotheses. Firstly, it clearly demonstrates the value of incorporating syntactic information into the document-level sentiment classifier, as the tree kernels (CON and D*) generally outperforms vector and sequence kernels (VK and S*). More importantly, it also shows the necessity of extracting appropriate substructures when using convolution kernels in our task: when using the dependency kernel (VK+DW), the result on lexicon guided substructures (Sub) outperforms the results on document, sentence, or randomly selected substructures, with statistical significance ($p<0.05$).

5 Conclusion and Future Work

We studied the impact of syntactic information on document-level sentiment classification using convolution kernels, and reduced the complexity of the kernels by extracting minimal high-impact substructures, guided by a polarity lexicon. Experiments

[1] A SubSet Tree is a structure that satisfies the constraint that grammatical rules cannot be broken, while a Partial Tree is a more general form of substructures obtained by the application of partial production rules of the grammar.

[2] available at http://disi.unitn.it/moschitti/

show that our method outperformed a bag-of-words baseline with a statistically significant gain of 1.45 absolute point in accuracy.

Our research focuses on identifying and using high-impact substructures for convolution kernels in document-level sentiment classification. We expect our method to be complementary with sophisticated methods used in state-of-the-art sentiment classification systems, which is to be explored in future work.

Acknowledgement

The authors were supported by 863 State Key Project No. 2006AA010108, the EuroMatrixPlus FP7 EU project (grant No 231720) and Science Foundation Ireland (Grant No. 07/CE/I1142). Part of the research was done while Zhaopeng Tu was visiting, and Yifan He was at the Centre for Next Generation Localisation (www.cngl.ie), School of Computing, Dublin City University. We thank the anonymous reviewers for their insightful comments. We are also grateful to Junhui Li for his helpful feedback.

References

Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, pages 30–38. Association for Computational Linguistics.

Razvan Bunescu and Raymond Mooney. 2005a. 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, oct. Association for Computational Linguistics.

Razvan Bunescu and Raymond Mooney. 2005b. Subsequence Kernels for Relation Extraction. In Y Weiss, B Sch o lkopf, and J Platt, editors, *Proceedings of the 19th Conference on Neural Information Processing Systems*, pages 171–178, Cambridge, MA. MIT Press.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Proceedings of Neural Information Processing Systems*, pages 625–632.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The stanford typed dependencies representation. In *Proceedings of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, Manchester, August.

Paul Ferguson, Neil O'Hare, Michael Davy, Adam Bermingham, Paraic Sheridan, Cathal Gurrin, and Alan F. Smeaton. 2009. Exploring the use of paragraph-level annotations for sentiment analysis of financial blogs. In *Proceedings of the Workshop on Opinion Mining and Sentiment Analysis*.

Richard Johansson and Alessandro Moschitti. 2010. Syntactic and semantic structure for opinion expression detection. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 67–76, Uppsala, Sweden, July.

Mahesh Joshi and Carolyn Penstein-Rose. 2009. Generalizing Dependency Features for Opinion Mining. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 313–316, Suntec, Singapore, jul. Suntec, Singapore.

Dan Klein and Christopher D Manning. 2003. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan, jul. Association for Computational Linguistics.

Moshe Koppel and Jonathan Schler. 2005. Using neutral examples for learning polarity. In *Proceedings of International Joint Conferences on Artificial Intelligence (IJCAI) 2005*, pages 1616–1616.

Steve Lawrence Kushal Dave and David Pennock. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of the 12th International Conference on World Wide Web*, pages 519–528, ACM. ACM.

Jingjing Liu and Stephanie Seneff. 2009. Review Sentiment Scoring via a Parse-and-Paraphrase Paradigm. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 161–169, Singapore, aug. Singapore.

Shotaro Matsumoto, Hiroya Takamura, and Manabu Okumura. 2005. Sentiment classification using word sub-sequences and dependency sub-trees. *Proceedings of PAKDD'05, the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, 3518/2005:21–32.

Alessandro Moschitti and Silvia Quarteroni. 2008. Kernels on Linguistic Structures for Answer Extraction. In *Proceedings of ACL-08: HLT, Short Papers*, pages 113–116, Columbus, Ohio, jun. Association for Computational Linguistics.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2008. Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224.

Alessandro Moschitti. 2006. Efficient Convolution Kernels for Dependency and Constituent Syntactic Trees. In *Proceedings of the 17th European Conference on Machine Learning*, pages 318–329, Berlin, Germany,

sep. Machine Learning: ECML 2006, 17th European Conference on Machine Learning, Proceedings.

Vincent Ng, Sajib Dasgupta, and S M Niaz Arifin. 2006. Examining the Role of Linguistic Knowledge Sources in the Automatic Identification and Classification of Reviews. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 611–618, Sydney, Australia, jul. Sydney, Australia.

Truc-Vien T Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. 2009. Convolution kernels on constituent, dependency and sequential structures for relation extraction. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1378–1387.

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*, pages 271–278, Barcelona, Spain, jun. Barcelona, Spain.

Michael Wiegand and Dietrich Klakow. 2010. Convolution Kernels for Opinion Holder Extraction. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 795–803, Los Angeles, California, jun. Los Angeles, California.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 347–354, Vancouver, British Columbia, Canada, oct. Association for Computational Linguistics.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 129–136, Association for Computational Linguistics. Association for Computational Linguistics.

Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 2006. A Composite Kernel to Extract Relations between Entities with Both Flat and Structured Features. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 825–832, Sydney, Australia, jul. Association for Computational Linguistics.

Exploiting Latent Information to Predict Diffusions of Novel Topics on Social Networks

Tsung-Ting Kuo[1]*, San-Chuan Hung[1], Wei-Shih Lin[1], Nanyun Peng[1], Shou-De Lin[1], Wei-Fen Lin[2]

[1]Graduate Institute of Networking and Multimedia, National Taiwan University, Taiwan

[2]MobiApps Corporation, Taiwan

*d97944007@csie.ntu.edu.tw

Abstract

This paper brings a marriage of two seemly unrelated topics, natural language processing (NLP) and social network analysis (SNA). We propose a new task in SNA which is to predict the diffusion of a new topic, and design a learning-based framework to solve this problem. We exploit the latent semantic information among users, topics, and social connections as features for prediction. Our framework is evaluated on real data collected from public domain. The experiments show 16% AUC improvement over baseline methods. The source code and dataset are available at http://www.csie.ntu.edu.tw/~d97944007/diffusion/

1 Background

The diffusion of information on social networks has been studied for decades. Generally, the proposed strategies can be categorized into two categories, model-driven and data-driven. The model-driven strategies, such as independent cascade model (Kempe et al., 2003), rely on certain manually crafted, usually intuitive, models to fit the diffusion data without using diffusion history. The data-driven strategies usually utilize learning-based approaches to predict the future propagation given historical records of prediction (Fei et al., 2011; Galuba et al., 2010; Petrovic et al., 2011). Data-driven strategies usually perform better than model-driven approaches because the past diffusion behavior is used during learning (Galuba et al., 2010).

Recently, researchers started to exploit content information in data-driven diffusion models (Fei et al., 2011; Petrovic et al., 2011; Zhu et al., 2011). However, most of the data-driven approaches assume that in order to train a model and predict the future diffusion of a topic, it is required to obtain historical records about how this topic has propagated in a social network (Petrovic et al., 2011; Zhu et al., 2011). We argue that such assumption does not always hold in the real-world scenario, and being able to forecast the propagation of novel or unseen topics is more valuable in practice. For example, a company would like to know which users are more likely to be the source of 'viva voce' of a *newly* released product for advertising purpose. A political party might want to estimate the potential degree of responses of a half-baked policy before deciding to bring it up to public. To achieve such goal, it is required to predict the future propagation behavior of a topic even *before* any actual diffusion happens on this topic (i.e., no historical propagation data of this topic are available). Lin et al. also propose an idea aiming at predicting the inference of implicit diffusions for novel topics (Lin et al., 2011). The main difference between their work and ours is that they focus on implicit diffusions, whose data are usually not available. Consequently, they need to rely on a model-driven approach instead of a data-driven approach. On the other hand, our work focuses on the prediction of explicit diffusion behaviors. Despite the fact that no diffusion data of novel topics is available, we can still design a data-driven approach taking advantage of some explicit diffusion data of known topics. Our experiments show that being able to utilize such information is critical for diffusion prediction.

2 The Novel-Topic Diffusion Model

We start by assuming an existing social network $G = (V, E)$, where V is the set of nodes (or user) v, and E is the set of link e. The set of topics is

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 344–348,
Jeju, Republic of Korea, 8-14 July 2012.

denoted as T. Among them, some are considered as novel topics (denoted as N), while the rest (R) are used as the training records. We are also given a set of diffusion records $D = \{d \mid d = (src, dest, t)\}$, where *src* is the source node (or diffusion source), *dest* is the destination node, and t is the topic of the diffusion that belongs to R but not N. We assume that diffusions cannot occur between nodes without direct social connection; any diffusion pair implies the existence of a link $e = (src, dest) \in E$. Finally, we assume there are sets of keywords or tags that relevant to *each* topic (including existing and novel topics). Note that the set of keywords for novel topics should be seen in that of existing topics. From these sets of keywords, we construct a topic-word matrix $TW = (P(word_j \mid topic_i))_{i,j}$ of which the elements stand for the conditional probabilities that a word appears in the text of a certain topic. Similarly, we also construct a user-word matrix $UW = (P(word_j \mid user_i))_{i,j}$ from these sets of keywords. Given the above information, the goal is to predict whether a given link is active (i.e., belongs to a diffusion link) for topics in N.

2.1 The Framework

The main challenge of this problem lays in that the past diffusion behaviors of new topics are missing. To address this challenge, we propose a supervised diffusion discovery framework that exploits the latent semantic information among users, topics, and their explicit / implicit interactions. Intuitively, four kinds of information are useful for prediction:

- *Topic information*: Intuitively, knowing the signatures of a topic (e.g., is it about politics?) is critical to the success of the prediction.
- *User information*: The information of a user such as the personality (e.g., whether this user is aggressive or passive) is generally useful.
- *User-topic interaction*: Understanding the users' preference on certain topics can improve the quality of prediction.
- *Global information*: We include some global features (e.g., topology info) of social network.

Below we will describe how these four kinds of information can be modeled in our framework.

2.2 Topic Information

We extract hidden topic category information to model *topic signature*. In particular, we exploit the Latent Dirichlet Allocation (LDA) method (Blei et al., 2003), which is a widely used topic modeling technique, to decompose the topic-word matrix TW into hidden topic categories:

$$TW = TH * HW$$

, where TH is a topic-hidden matrix, HW is hidden-word matrix, and h is the manually-chosen parameter to determine the size of hidden topic categories. TH indicates the distribution of each topic to hidden topic categories, and HW indicates the distribution of each lexical term to hidden topic categories. Note that TW and TH include both existing and novel topics. We utilize $TH_{t,*}$, the row vector of the topic-hidden matrix TH for a topic t, as a feature set. In brief, we apply LDA to extract the topic-hidden vector $TH_{t,*}$ to model *topic signature (TG)* for both existing and novel topics.

Topic information can be further exploited. To predict whether a novel topic will be propagated through a link, we can first enumerate the existing topics that have been propagated through this link. For each such topic, we can calculate its similarity with the new topic based on the hidden vectors generated above (e.g., using cosine similarity between feature vectors). Then, we sum up the similarity values as a new feature: *topic similarity (TS)*. For example, a link has previously propagated two topics for a total of three times {ACL, KDD, ACL}, and we would like to know whether a new topic, EMNLP, will propagate through this link. We can use the topic-hidden vector to generate the similarity values between EMNLP and the other topics (e.g., {0.6, 0.4, 0.6}), and then sum them up (1.6) as the value of TS.

2.3 User Information

Similar to topic information, we extract latent personal information to model *user signature* (the users are anonymized already). We apply LDA on the user-word matrix UW:

$$UW = UM * MW$$

, where UM is the user-hidden matrix, MW is the hidden-word matrix, and m is the manually-chosen size of hidden user categories. UM indicates the distribution of each user to the hidden user categories (e.g., age). We then use $UM_{u,*}$, the row vector of UM for the user u, as a feature set. In brief, we apply LDA to extract the user-hidden vector $UM_{u,*}$ for both source and destination nodes of a link to model *user signature (UG)*.

2.4 User-Topic Interaction

Modeling user-topic interaction turns out to be non-trivial. It is not useful to exploit latent semantic analysis directly on the user-topic matrix $UR = UQ * QR$, where UR represents *how many times each user is diffused for existing topic R* ($R \in T$), because UR does not contain information of novel topics, and neither do UQ and QR. Given no propagation record about novel topics, we propose a method that allows us to still extract implicit user-topic information. First, we extract from the matrix TH (described in Section 2.2) a subset RH that contains only information about existing topics. Next we apply left division to derive another user-hidden matrix UH:

$$UH = (RH \setminus UR^{T})^{T} = ((RH^{T} RH)^{-1} RH^{T} UR^{T})^{T}$$

Using left division, we generate the UH matrix using existing topic information. Finally, we exploit $UH_{u,*}$, the row vector of the user-hidden matrix UH for the user u, as a feature set.

Note that novel topics were included in the process of learning the hidden topic categories on RH; therefore the features learned here do implicitly utilize some latent information of novel topics, which is not the case for UM. Experiments confirm the superiority of our approach. Furthermore, our approach ensures that the hidden categories in topic-hidden and user-hidden matrices are identical. Intuitively, our method directly models the user's preference to topics' signature (e.g., how capable is this user to propagate topics in politics category?). In contrast, the UM mentioned in Section 2.3 represents the users' signature (e.g., aggressiveness) and has nothing to do with their opinions on a topic. In short, we obtain the user-hidden probability vector $UH_{u,*}$ as a feature set, which models *user preferences to latent categories (UPLC).*

2.5 Global Features

Given a candidate link, we can extract global social features such as *in-degree (ID)* and *out-degree (OD).* We tried other features such as PageRank values but found them not useful. Moreover, we extract the *number of distinct topics (NDT)* for a link as a feature. The intuition behind this is that the more distinct topics a user has diffused to another, the more likely the diffusion will happen for novel topics.

2.6 Complexity Analysis

The complexity to produce each feature is as below:

(1) *Topic information*: $O(I * |T| * h * B_t)$ for LDA using Gibbs sampling, where I is # of the iterations in sampling, $|T|$ is # of topics, and B_t is the average # of tokens in a topic.

(2) *User information*: $O(I * |V| * m * B_u)$, where $|V|$ is # of users, and B_u is the average # of tokens for a user.

(3) *User-topic interaction*: the time complexity is $O(h^3 + h^2 * |T| + h * |T| * |V|)$.

(4) *Global features*: $O(|D|)$, where $|D|$ is # of diffusions.

3 Experiments

For evaluation, we try to use the diffusion records of old topics to predict whether a diffusion link exists between two nodes given a new topic.

3.1 Dataset and Evaluation Metric

We first identify 100 most popular topic (e.g., earthquake) from the Plurk micro-blog site between 01/2011 and 05/2011. Plurk is a popular micro-blog service in Asia with more than 5 million users (Kuo et al., 2011). We manually separate the 100 topics into 7 groups. We use topic-wise 4-fold cross validation to evaluate our method, because there are only 100 available topics. For each group, we select 3/4 of the topics as training and 1/4 as validation.

The positive diffusion records are generated based on the post-response behavior. That is, if a person x posts a message containing one of the selected topic t, and later there is a person y responding to this message, we consider a diffusion of t has occurred from x to y (i.e., *(x, y, t)* is a positive instance). Our dataset contains a total of 1,642,894 positive instances out of 100 distinct topics; the largest and smallest topic contains 303,424 and 2,166 diffusions, respectively. Also, the same amount of negative instances for each topic (totally 1,642,894) is sampled for binary classification (similar to the setup in KDD Cup 2011 Track 2). The negative links of a topic t are sampled randomly based on the absence of responses for that given topic.

The underlying social network is created using the post-response behavior as well. We assume there is an acquaintance link between x and y if and

only if x has responded to y (or vice versa) on at least one topic. Eventually we generated a social network of 163,034 nodes and 382,878 links. Furthermore, the sets of keywords for each topic are required to create the *TW* and *UW* matrices for latent topic analysis; we simply extract the content of posts and responses for each topic to create both matrices. We set the hidden category number $h = m = 7$, which is equal to the number of topic groups.

We use area under ROC curve (AUC) to evaluate our proposed framework (Davis and Goadrich, 2006); we rank the testing instances based on their likelihood of being positive, and compare it with the ground truth to compute AUC.

3.2 Implementation and Baseline

After trying many classifiers and obtaining similar results for all of them, we report only results from LIBLINEAR with c=0.0001 (Fan et al., 2008) due to space limitation. We remove stop-words, use SCWS (Hightman, 2012) for tokenization, and MALLET (McCallum, 2002) and GibbsLDA++ (Phan and Nguyen, 2007) for LDA.

There are three baseline models we compare the result with. First, we simply use the total number of existing diffusions among all topics between two nodes as the single feature for prediction. Second, we exploit the independent cascading model (Kempe et al., 2003), and utilize the normalized total number of diffusions as the propagation probability of each link. Third, we try the heat diffusion model (Ma et al., 2008), set initial heat proportional to out-degree, and tune the diffusion time parameter until the best results are obtained. Note that we did not compare with any data-driven approaches, as we have not identified one that can predict diffusion of novel topics.

3.3 Results

The result of each model is shown in Table 1. All except two features outperform the baseline. The best single feature is *TS*. Note that *UPLC* performs better than *UG*, which verifies our hypothesis that maintaining the same hidden features across different LDA models is better. We further conduct experiments to evaluate different combinations of features (Table 2), and found that the best one (*TS + ID + NDT*) results in about 16% improvement over the baseline, and outperforms the combination of all features. As stated in (Witten et al., 2011), adding useless features may cause the performance of classifiers to deteriorate. Intuitively, *TS* captures both latent topic and historical diffusion information, while *ID* and *NDT* provide complementary social characteristics of users.

Method	Feature	AUC
Baseline	Existing Diffusion	58.25%
	Independent Cascade	51.53%
	Heat Diffusion	56.08%
Learning	**T**opic Sig**n**ature (*TG*)	50.80%
	Topic **S**imilarity (*TS*)	**69.93%**
	User Sig**n**ature (*UG*)	56.59%
	User **P**references to **L**atent **C**ategories (*UPLC*)	61.33%
	In-**d**egree (*ID*)	65.55%
	Out-**d**egree (*OD*)	59.73%
	Number of **D**istinct **T**opics (*NDT*)	55.42%

Table 1: Single-feature results.

Method	Feature	AUC
Baseline	Existing Diffusion	58.25%
Learning	ALL	65.06%
	TS + UPLC + ID + NDT	67.67%
	TS + UPLC + ID	64.80%
	TS + UPLC + NDT	66.01%
	TS + ID + NDT	**73.95%**
	UPLC + ID + NDT	67.24%

Table 2: Feature combination results.

4 Conclusions

The main contributions of this paper are as below:

1. We propose a novel task of predicting the diffusion of unseen topics, which has wide applications in real-world.
2. Compared to the traditional model-driven or content-independent data-driven works on diffusion analysis, our solution demonstrates how one can bring together ideas from two different but promising areas, NLP and SNA, to solve a challenging problem.
3. Promising experiment result (74% in AUC) not only demonstrates the usefulness of the proposed models, but also indicates that predicting diffusion of unseen topics without historical diffusion data is feasible.

Acknowledgments

This work was also supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC 100-2911-I-002-001, and 101R7501.

References

David M. Blei, Andrew Y. Ng & Michael I. Jordan. 2003. Latent dirichlet allocation. J. Mach. Learn. Res., 3.993-1022.

Jesse Davis & Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. Proceedings of the 23rd international conference on Machine learning, Pittsburgh, Pennsylvania.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang & Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. J. Mach. Learn. Res., 9.1871-74.

Hongliang Fei, Ruoyi Jiang, Yuhao Yang, Bo Luo & Jun Huan. 2011. Content based social behavior prediction: a multi-task learning approach. Proceedings of the 20th ACM international conference on Information and knowledge management, Glasgow, Scotland, UK.

Wojciech Galuba, Karl Aberer, Dipanjan Chakraborty, Zoran Despotovic & Wolfgang Kellerer. 2010. Outtweeting the twitterers - predicting information cascades in microblogs. Proceedings of the 3rd conference on Online social networks, Boston, MA.

Hightman. 2012. Simple Chinese Words Segmentation (SCWS).

David Kempe, Jon Kleinberg & Eva Tardos. 2003. Maximizing the spread of influence through a social network. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, Washington, D.C.

Tsung-Ting Kuo, San-Chuan Hung, Wei-Shih Lin, Shou-De Lin, Ting-Chun Peng & Chia-Chun Shih. 2011. Assessing the Quality of Diffusion Models Using Real-World Social Network Data. Conference on Technologies and Applications of Artificial Intelligence, 2011.

C.X. Lin, Q.Z. Mei, Y.L. Jiang, J.W. Han & S.X. Qi. 2011. Inferring the Diffusion and Evolution of Topics in Social Communities. Proceedings of the IEEE International Conference on Data Mining, 2011.

Hao Ma, Haixuan Yang, Michael R. Lyu & Irwin King. 2008. Mining social networks using heat diffusion processes for marketing candidates selection. Proceeding of the 17th ACM conference on Information and knowledge management, Napa Valley, California, USA.

Andrew Kachites McCallum. 2002. MALLET: A Machine Learning for Language Toolkit.

Sasa Petrovic, Miles Osborne & Victor Lavrenko. 2011. RT to Win! Predicting Message Propagation in Twitter. International AAAI Conference on Weblogs and Social Media, 2011.

Xuan-Hieu Phan & Cam-Tu Nguyen. 2007. GibbsLDA++: A C/C++ implementation of latent Dirichlet allocation (LDA).

Ian H. Witten, Eibe Frank & Mark A. Hall. 2011. Data Mining: Practical machine learning tools and techniques. San Francisco: Morgan Kaufmann Publishers Inc.

Jiang Zhu, Fei Xiong, Dongzhen Piao, Yun Liu & Ying Zhang. 2011. Statistically Modeling the Effectiveness of Disaster Information in Social Media. Proceedings of the 2011 IEEE Global Humanitarian Technology Conference.

Sentence Compression with Semantic Role Constraints

Katsumasa Yoshikawa
Precision and Intelligence Laboratory,
Tokyo Institute of Technology, Japan
IBM Research-Tokyo, IBM Japan, Ltd.
katsumasay@gmail.com

Ryu Iida
Department of Computer Science,
Tokyo Institute of Technology, Japan
ryu-i@cl.cs.titech.ac.jp

Tsutomu Hirao
NTT Communication Science Laboratories,
NTT Corporation, Japan
hirao.tsutomu@lab.ntt.co.jp

Manabu Okumura
Precision and Intelligence Laboratory,
Tokyo Institute of Technology, Japan
oku@lr.pi.titech.ac.jp

Abstract

For sentence compression, we propose new semantic constraints to directly capture the relations between a predicate and its arguments, whereas the existing approaches have focused on relatively shallow linguistic properties, such as lexical and syntactic information. These constraints are based on semantic roles and superior to the constraints of syntactic dependencies. Our empirical evaluation on the Written News Compression Corpus (Clarke and Lapata, 2008) demonstrates that our system achieves results comparable to other state-of-the-art techniques.

1 Introduction

Recent work in document summarization do not only extract sentences but also compress sentences. Sentence compression enables summarizers to reduce the redundancy in sentences and generate informative summaries beyond the extractive summarization systems (Knight and Marcu, 2002). Conventional approaches to sentence compression exploit various linguistic properties based on lexical information and syntactic dependencies (McDonald, 2006; Clarke and Lapata, 2008; Cohn and Lapata, 2008; Galanis and Androutsopoulos, 2010).

In contrast, our approach utilizes another property based on semantic roles (SRs) which improves weaknesses of syntactic dependencies. Syntactic dependencies are not sufficient to compress some complex sentences with coordination, with passive voice, and with an auxiliary verb. Figure 1 shows an example with a coordination structure. [1]

[1]This example is from Written News Compression Corpus (http://jamesclarke.net/research/resources).

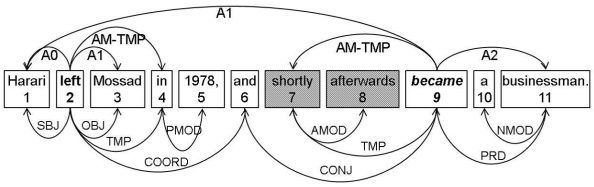


Figure 1: Semantic Role vs. Dependency Relation

In this example, a SR labeler annotated that *Harari* is an A0 argument of *left* and an A1 argument of *became*. *Harari* is syntactically dependent on *left* – *SBJ(left-2, Harari-1)*. However, *Harari* is not dependent on *became* and we are hence unable to utilize a dependency relation between *Harari* and *became* directly. SRs allow us to model the relations between a predicate and its arguments in a direct fashion.

SR constraints are also advantageous in that we can compress sentences with semantic information. In Figure 1, *became* has three arguments, *Harari* as A1, *businessman* as A2, and *shortly afterward* as AM-TMP. As shown in this example, *shortly afterword* can be omitted (shaded boxes). In general, modifier arguments like AM-TMP or AM-LOC are more likely to be reduced than complement cases like A0-A4. We can implement such properties by SR constraints.

Liu and Gildea (2010) suggests that SR features contribute to generating more readable sentence in machine translation. We expect that SR features also help our system to improve readability in sentence compression and summarization.

2 Why are Semantic Roles Useful for Compressing Sentences?

Before describing our system, we show the statistics in terms of predicates, arguments and their rela-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 349–353,
Jeju, Republic of Korea, 8-14 July 2012.

Label	In Compression / Total	Ratio
A0	1454 / 1607	0.905
A1	1916 / 2208	0.868
A2	427 / 490	0.871
AM-TMP	261 / 488	0.535
AM-LOC	134 / 214	0.626
AM-ADV	115 / 213	0.544
AM-DIS	8 / 85	0.094

Table 1: Statistics of Arguments in Compression

tions in the Written News Compression (WNC) Corpus. It has 82 documents (1,629 sentences). We divided them into three: 55 documents are used for training (1106 sentences); 10 for development (184 sentences); 17 for testing (339 sentences).

Our investigation was held in training data. There are 3137 verbal predicates and 7852 unique arguments. We performed SR labeling by LTH (Johansson and Nugues, 2008), an SR labeler for CoNLL-2008 shared task. Based on the SR labels annotated by LTH, we investigated that, for all predicates in compression, how many their arguments were also in. Table 1 shows the survival ratio of main arguments in compression. Labels A0, A1, and A2 are complement case roles and over 85% of them survive with their predicates. On the other hand, for modifier arguments (AM-*X*), survival ratios are down to lower than 65%. Our SR constraints implement the difference of survival ratios by SR labels. Note that dependency labels SBJ and OBJ generally correspond to SR labels A0 and A1, respectively. But their total numbers are 777 / 919 (SBJ) and 918 / 1211 (OBJ) and much fewer than A0 and A1 labels. Thus, SR labels can connect much more arguments to their predicates.

3 Approach

This section describes our new approach to sentence compression. In order to introduce rich syntactic and semantic constraints to a sentence compression model, we employ Markov Logic (Richardson and Domingos, 2006). Since Markov Logic supports both soft and hard constraints, we can implement our SR constraints in simple and direct fashion. Moreover, implementations of learning and inference methods are already provided in existing Markov Logic interpreters such as *Alchemy* [2] and *Markov thebeast*. [3] Thus, we can focus our effort on building a set of formulae called Markov Logic Network (MLN). So, in this section, we describe our proposed MLN in detail.

3.1 Proposed Markov Logic Network

First, let us define our MLN predicates. We summarize the MLN predicates in Table 2. We have only one *hidden* MLN predicate, $inComp(i)$ which models the decision we need to make: whether a token i is in compression or not. The other MLN predicates are called *observed* which provide features. With our MLN predicates defined, we can now go on to incorporate our intuition about the task using weighted first-order logic formulae. We define SR constraints and the other formulae in Sections 3.1.1 and 3.1.2, respectively.

3.1.1 Semantic Role Constraints

Semantic role labeling generally includes the three subtasks: predicate identification; argument role labeling; sense disambiguation. Our model exploits the results of predicate identification and argument role labeling. [4] $pred(i)$ and $role(i, j, r)$ indicate the results of predicate identification and role labeling, respectively.

First, the formula describing a local property of a predicate is

$$pred(i) \Rightarrow \text{inComp}(i) \quad (1)$$

which denotes that, if token i is a predicate then i is in compression. A formula with exact one *hidden* predicate is called *local* formula.

A predicate is not always in compression. The formula reducing some predicates is

$$pred(i) \wedge height(i, +n) \Rightarrow \neg\text{inComp}(i) \quad (2)$$

which implies that a predicate i is not in compression with n height in a dependency tree. Note the + notation indicates that the MLN contains one instance of the rule, with a separate weight, for each assignment of the variables with a plus sign.

As mentioned earlier, our SR constraints model the difference of the survival rate of role labels in compression. Such SR constraints are encoded as:

$$role(i, j, +r) \wedge \text{inComp}(i) \Rightarrow \text{inComp}(j) \quad (3)$$

$$role(i, j, +r) \wedge \neg\text{inComp}(i) \Rightarrow \neg\text{inComp}(j) \quad (4)$$

which represent that, if a predicate i is (not) in compression, then its argument j is (not) also in with

[2] http://alchemy.cs.washington.edu/

[3] http://code.google.com/p/thebeast/

[4] Sense information is too sparse because the size of the WNC Corpus is not big enough.

predicate	definition
inComp(i)	Token i is in compression
$pred(i)$	Token i is a predicate
$role(i, j, r)$	Token i has an argument j with role r
$word(i, w)$	Token i has word w
$pos(i, p)$	Token i has Pos tag p
$dep(i, j, d)$	Token i is dependent on token j with dependency label d
$path(i, j, l)$	Tokens i and j has syntactic path l
$height(i, n)$	Token i has height n in dependency tree

Table 2: MLN Predicates

role r. These formulae are called *global* formulae because they have more than two *hidden* MLN predicates. With global formulae, our model makes two decisions at a time. When considering the example in Figure 1, Formula (3) will be grounded as:

$$role(9, 1, \text{A0}) \wedge \text{inComp}(9) \Rightarrow \text{inComp}(1) \quad (5)$$

$$role(9, 7, \text{AM-TMP}) \wedge \text{inComp}(9) \Rightarrow \text{inComp}(7). \quad (6)$$

In fact, Formula (5) gains a higher weight than Formula (6) by learning on training data. As a result, our system gives "1-*Harari*" more chance to survive in compression. We also add some extensions of Formula (3) combined with $dep(i, j, +d)$ and $path(i, j, +l)$ which enhance SR constraints. Note, all our SR constraints are "predicate-driven" (only $\Rightarrow$ not $\Leftrightarrow$ as in Formula (13)). Because an argument is usually related to multiple predicates, it is difficult to model "argument-driven" formula.

3.1.2 Lexical and Syntactic Features

For lexical and syntactic features, we mainly refer to the previous work (McDonald, 2006; Clarke and Lapata, 2008). The first two formulae in this section capture the relation of the tokens with their lexical and syntactic properties. The formula describing such a local property of a word form is

$$word(i, +w) \Rightarrow \text{inComp}(i) \quad (7)$$

which implies that a token i is in compression with a weight that depends on the word form.

For part-of-speech (POS), we add unigram and bigram features with the formulae,

$$pos(i, +p) \Rightarrow \text{inComp}(i) \quad (8)$$

$$pos(i, +p_1) \wedge pos(i+1, +p_2) \Rightarrow \text{inComp}(i). \quad (9)$$

POS features are often more reasonable than word form features to combine with the other properties. The formula,

$$pos(i, +p) \wedge height(i, +n) \Rightarrow \text{inComp}(i). \quad (10)$$

is a combination of POS features and a height in a dependency tree.

The next formula combines POS bigram features with dependency relations.

$$pos(i, +p_1) \wedge pos(j, +p_2) \wedge dep(i, j, +d) \Rightarrow \text{inComp}(i). \quad (11)$$

Moreover, our model includes the following global formulae,

$$dep(i, j, +d) \wedge \text{inComp}(i) \Rightarrow \text{inComp}(j) \quad (12)$$

$$dep(i, j, +d) \wedge \text{inComp}(i) \Leftrightarrow \text{inComp}(j) \quad (13)$$

which enforce the consistencies between head and modifier tokens. Formula (12) represents that if we include a head token in compression then its modifier must also be included. Formula (13) ensures that head and modifier words must be simultaneously kept in compression or dropped. Though Clarke and Lapata (2008) implemented these dependency constraints by ILP, we implement them by soft constraints of MLN. Note that Formula (12) expresses the same properties as Formula (3) replacing $dep(i, j, +d)$ by $role(i, j, +r)$.

4 Experiment and Result

4.1 Experimental Setup

Our experimental setting follows previous work (Clarke and Lapata, 2008). As stated in Section 2, we employed the WNC Corpus. For preprocessing, we performed POS tagging by stanford-tagger. [5] and dependency parsing by MST-parser (McDonald et al., 2005). In addition, LTH [6] was exploited to perform both dependency parsing and SR labeling. We implemented our model by Markov Thebeast with Gurobi optimizer. [7]

Our evaluation consists of two types of automatic evaluations. The first evaluation is dependency based evaluation same as Riezler et al. (2003). We performed dependency parsing on gold data and system outputs by RASP. [8] Then we calculated precision, recall, and F1 for the set of $label(head, modifier)$.

In order to demonstrate how well our SR constraints keep correct predicate-argument structures in compression, we propose SRL based evaluation. We performed SR labeling on gold data

[5] http://nlp.stanford.edu/software/tagger.shtml

[6] http://nlp.cs.lth.se/software/semantic_parsing:_propbank_nombank_frames

[7] http://www.gurobi.com/

[8] http://www.informatics.susx.ac.uk/research/groups/nlp/rasp/

Original	[$_{A0}$ They] [$_{pred}$ say] [$_{A1}$ the refugees will enhance productivity and economic growth].
MLN with SRL	[$_{A0}$ They] [$_{pred}$ say] [$_{A1}$ the refugees will enhance growth].
Gold Standard	[$_{A1*}$ the refugees will enhance productivity and growth].
Original	[$_{A0}$ A Ł16.1m dam] [$_{AM-MOD}$ will] [$_{pred}$ hold] back [$_{A1}$ a 2.6-mile-long artificial lake to be known as the Roadford Reservoir].
MLN with SRL	[$_{A0}$ A dam] will [$_{pred}$ hold] back [$_{A1}$ a artificial lake to be known as the Roadford Reservoir].
Gold Standard	[$_{A0}$ A Ł16.1m dam] [$_{AM-MOD}$ will] [$_{pred}$ hold back [$_{A1}$ a 2.6-mile-long Roadford Reservoir].

Table 4: Analysis of Errors

Model	CompR	F1-Dep	F1-SRL
McDonald	73.6%	38.4%	49.9%
MLN w/o SRL	68.3%	51.3%	57.2%
MLN with SRL	73.1%	58.9%	64.1%
Gold Standard	73.3%	–	–

Table 3: Results of Sentence Compression

and system outputs by LTH. Then we calculated precision, recall, and F1 value for the set of *role*(*predicate*, *argument*).

The training time of our MLN model are approximately 8 minutes on all training data, with 3.1GHz Intel Core i3 CPU and 4G memory. While the prediction can be done within 20 seconds on the test data.

4.2 Results

Table 3 shows the results of our compression models by compression rate (CompR), dependency-based F1 (F1-Dep), and SRL-based F1 (F1-SRL). In our experiment, we have three models. **McDonald** is a re-implementation of McDonald (2006). Clarke and Lapata (2008) also re-implemented McDonald's model with an ILP solver and experimented it on the WNC Corpus. [9] **MLN with SRL** and **MLN w/o SRL** are our Markov Logic models with and without SR Constraints, respectively.

Note our three models have no constraint for the length of compression. Therefore, we think the compression rate of the better system should get closer to that of human compression. In comparison between MLN models and McDonald, the former models outperform the latter model on both F1-Dep and F1-SRL. Because MLN models have global constraints and can generate syntactically correct sentences.

Our concern is how a model with SR constraints is superior to a model without them. MLN with SRL outperforms MLN without SRL with a 7.6 points margin (F1-Dep). The compression rate of MLN with SRL goes up to 73.1% and gets close to that of gold standard. SRL-based evaluation also shows that SR constraints actually help extract correct predicate-argument structures. These results are promising to improve readability.

It is difficult to directly compare our results with those of state-of-the-art systems (Cohn and Lapata, 2009; Clarke and Lapata, 2010; Galanis and Androutsopoulos, 2010) since they have different testing sets and the results with different compression rates. However, though our MLN model with SR constraints utilizes no large-scale data, it is the only model which achieves close on 60% in F1-Dep.

4.3 Error Analysis

Table 4 indicates two critical examples which our SR constraints failed to compress correctly. For the first example, our model leaves an argument with its predicate because our SR constraints are "predicate-driven". In addition, "say" is the main verb in this sentence and hard to be deleted due to the syntactic significance.

The second example in Table 4 requires to identify a coreference relation between *artificial lake* and *Roadford Reservour*. We consider that discourse constraints (Clarke and Lapata, 2010) help our model handle these cases. Discourse and coreference information enable our model to select important arguments and their predicates.

5 Conclusion

In this paper, we proposed new semantic constraints for sentence compression. Our model with global constraints of semantic roles selected correct predicate-argument structures and successfully improved performance of sentence compression.

As future work, we will compare our model with the other state-of-the-art systems. We will also investigate the correlation between readability and SRL-based score by manual evaluations. Furthermore, we would like to combine discourse constraints with SR constraints.

[9] Clarke's re-implementation got 60.1% for CompR and 36.0%pt for F1-Dep

References

James Clarke and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31(1):399–429.

James Clarke and Mirella Lapata. 2010. Discourse constraints for document compression. *Computational Linguistics*, 36(3):411–441.

Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 137–144. Association for Computational Linguistics.

Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Dimitrios Galanis and Ion Androutsopoulos. 2010. An extractive supervised two-stage method for sentence compression. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 885–893, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic-semantic analysis with propbank and nombank. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 183–187. Association for Computational Linguistics.

Kevin Knight and Daniel Marcu. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

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.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan 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*, HLT '05, pages 523–530, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ryan McDonald. 2006. Discriminative sentence compression with soft syntactic evidence. In *Proceedings of EACL*, pages 297–304.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning*, 62(1-2):107–136.

Stefan Riezler, Tracy H. King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 118–125. Association for Computational Linguistics.

Fully Abstractive Approach to Guided Summarization

Pierre-Etienne Genest, Guy Lapalme
RALI-DIRO
Université de Montréal
P.O. Box 6128, Succ. Centre-Ville
Montréal, Québec
Canada, H3C 3J7
{genestpe,lapalme}@iro.umontreal.ca

Abstract

This paper shows that full abstraction can be accomplished in the context of guided summarization. We describe a work in progress that relies on Information Extraction, statistical content selection and Natural Language Generation. Early results already demonstrate the effectiveness of the approach.

1 Introduction

In the last decade, automatic text summarization has been dominated by extractive approaches that rely purely on shallow statistics. In the latest evaluation campaign of the Text Analysis Conference[1] (TAC), the top systems were considered only "barely acceptable" by human assessment (Owczarzak and Dang, 2011). The field is also getting saturated near what appears to be a ceiling in performance. Systems that claim to be very different from one another have all become statistically indistinguishable in evaluation results. An experiment (Genest et al., 2009) found a performance ceiling to pure sentence extraction that is very low compared to regular (abstractive) human summaries, but not that much better than the current best automatic systems.

Abstractive summarization has been explored to some extent in recent years: sentence compression (Knight and Marcu, 2000) (Cohn and Lapata, 2009), sentence fusion (Barzilay and McKeown, 2005) or revision (Tanaka et al., 2009), and a generation-based approach that could be called sentence splitting (Genest and Lapalme, 2011). They are all rewriting techniques based on syntactical analysis, offering little improvement over extractive methods in the content selection process.

We believe that a fully abstractive approach with a separate process for the analysis of the text, the content selection, and the generation of the summary has the most potential for generating summaries at a level comparable to human. For the foreseeable future, we think that such a process for full abstraction is impossible in the general case, since it is almost equivalent to perfect text understanding. In specific domains, however, an approximation of full abstraction is possible.

This paper shows that full abstraction can be accomplished in the context of guided summarization. We propose a methodology that relies on Information Extraction and Natural Language Generation, and discuss our early results.

2 Guided Summarization

The stated goal of the guided summarization task at TAC is to motivate a move towards abstractive approaches. It is an oriented multidocument summarization task in which a category is attributed to a cluster of 10 source documents to be summarized in 100 words or less. There are five categories: Accidents and Natural Disasters, Attacks, Health and Safety, Endangered Resources, and Investigations/Trials. Each category is associated with a list of aspects to address in the summary. Figure 1 shows the aspects for the Attacks category. We use this specification of categories and aspects to accomplish domain-specific summarization.

[1] www.nist.gov/tac

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 354–358,
Jeju, Republic of Korea, 8-14 July 2012.

2.1 WHAT: what happened
2.2 WHEN: date, time, other temporal placement markers
2.3 WHERE: physical location
2.4 PERPETRATORS: individuals or groups responsible for the attack
2.5 WHY: reasons for the attack
2.6 WHO_AFFECTED: casualties (death, injury), or individuals otherwise negatively affected
2.7 DAMAGES: damages caused by the attack
2.8 COUNTERMEASURES: countermeasures, rescue efforts, prevention efforts, other reactions

Figure 1: Aspects for TAC's guided summarization task, category 2: Attacks

3 Fully Abstractive Approach

Guided summarization categories and aspects define an information need, and using Information Extraction (IE) seems appropriate to address it. The idea to use an IE system for summarization can be traced back to the FRUMP system (DeJong, 1982), which generates brief summaries about various kinds of stories; (White et al., 2001) also wrote abstractive summaries using the output of an IE system applied to events such as natural disasters. In both cases, the end result is a generated summary from the information available. A lot of other work has instead used IE to improve the performance of extraction-based systems, like (Barzilay and Lee, 2004) and (Ji et al., 2010).

What is common to all these approaches is that the IE system is designed for a specific purpose, separate from summarization. However, to properly address each aspect requires a system designed specifically for that task. To our knowledge, tailoring IE to the needs of abstractive summarization has not been done before. Our methodology uses a rule-based, custom-designed IE module, integrated with Content Selection and Generation in order to write short, well-written abstractive summaries.

Before tackling these, we perform some preprocessing on the cluster of documents. It includes: cleaning up and normalization of the input using regular expressions, sentence segmentation, tokenization and lemmatization using GATE (Cunningham et al., 2002), syntactical parsing and dependency parsing (collapsed) using the Stanford Parser (de Marneffe et al., 2006), and Named Entity Recognition using Stanford NER (Finkel et al., 2005). We have also developed a date resolution engine that focuses on days of the week and relative terms.

3.1 Information Extraction

Our architecture is based on *Abstraction Schemes*. An abstraction scheme consists of IE rules, content selection heuristics and one or more generation patterns, all created by hand. Each abstraction scheme is designed to address a theme or subcategory. Thus, rules that extract information for the same aspect within the same scheme will share a similar meaning. An abstraction scheme aims to answer one or more aspects of its category, and more than one scheme can be linked to the same aspect.

Figure 2 shows two of the schemes that we have created. For the scheme **killing**, the IE rules would match X as the perpetrator and Y as a victim for all of the following phrases: X `killed` Y, Y `was assassinated by` X, and `the murder of` X `by` Y. Other schemes have similar structure and purpose, such as **wounding**, **abducting**, **damaging** and **destroying**. To create extraction rules for a scheme, we must find several verbs and nouns sharing a similar meaning and identify the syntactical position of the roles we are interested in. Three resources have helped us in designing extraction rules: a thesaurus to find semantically related nouns and verbs; VerbNet (Kipper et al., 2006), which provides amongst other things the semantic roles of the syntactical dependents of verbs; and a hand-crafted list of aspect-relevant word stems provided by the team that made CLASSY (Conroy et al., 2010).

Schemes and their extraction rules can also be quite different from this first example, as shown with the scheme **event**. This scheme gathers the basic information about the attack event: WHAT category of attack, WHEN and WHERE it occurred. A list of *key word*s is used to identify words that imply an attack event, while a list of EVENT NOUNs is used to identify specifically words that refer to a type of attack.

Scheme: killing

Information Extraction	SUBJ(`kill`, X) → WHO(X) OBJ(`kill`, Y) → WHO_AFFECTED(Y) SUBJ(`assassinate`, X) → WHO(X) OBJ(`assassinate`, Y) → WHO_AFFECTED(Y) ⋮ PREP_OF(`murder`, Y) → WHO_AFFECTED(Y) PREP_BY(`murder`, X) → WHO(X) ⋮
Content Selection	Select best candidates for *kill verb*, WHO(X) and WHO_AFFECTED(Y)
Generation	X *kill verb* Y

Scheme: event

Information Extraction	PREP_IN(*key word*, X), LOCATION(X) → WHERE(X) PREP_IN(*key word*, X), ORGANIZATION(X) → WHERE(X) PREP_AT(*key word*, X), LOCATION(X) → WHERE(X) PREP_AT(*key word*, X), ORGANIZATION(X) → WHERE(X) DEP(*key word*, Y), DATE(Y) → WHEN(Y) EVENT NOUN(Z) → WHAT(Z)
Content Selection	Select best candidates for `at` or `in`, WHERE(X), WHEN(Y) and WHAT(Z)
Generation	`On` Y, Z `occurred at/in` X

Figure 2: Abstraction schemes **killing** and **event**. The information extraction rules translate preprocessing annotations into candidate answers for a specific aspect. Content selection determines which candidate will be included in the generated sentence for each aspect. Finally, a pattern is used to determine the structure of the generated sentence. Notation: `word or lemma`, variable, *group of words*, PREDICATE OR ASPECT. Note that the predicate DEP matches any syntactical dependency and that *key words* refer to a premade list of category-relevant verbs and nouns.

3.2 Content Selection

A large number of candidates are found by the IE rules for each aspect. The content selection module selects the best ones and sends them to the generation module. The basic heuristic is to select the candidate most often mentioned for an aspect, and similarly for the choice of a preposition or a verb for generation. More than one candidate may be selected for the aspect WHO_AFFECTED, the victims of the attack. Several heuristics are used to avoid redundancies and uninformative answers.

News articles may contain references to more than one event of a given category, but our summaries describe only one. To avoid mixing candidates from two different event instances that might appear in the same cluster of documents, we rely on dates. The ancestors of a date in the dependency tree are associated with that date, and excluded from the summary if the main event occurs on a different date.

3.3 Generation

The text of a summary must be fluid and feel natural, while being straightforward and concise. From our observation of human-written summaries, it also does not require a great deal of originality to be considered excellent by human standards. Thus, we have designed straightforward generation patterns for each scheme. They are implemented using the SimpleNLG realizer (Gatt and Reiter, 2009), which takes a sentence structure and words in their root form as input and gives a sentence with resolved agreements and sentence markers as output. The greatest difficulty in the structure is in realizing noun phrases. The content selection module selects a lemma that should serve as noun phrase head, and its number, modifiers and specifier must be determined during generation. Frequencies and heuristics are again used to identify appropriate modifiers, this time from all those used with that head within the source documents. We apply the constraint that the

On April 20, 1999, a massacre occurred at Columbine High School.
Two student gunmen killed 12 students, a teacher and themselves.

On November 2, 2004, a brutal murder occurred in Amsterdam.
A gunman stabbed and shot Dutch filmmaker Theo van Gogh.
A policeman and the suspect were wounded.

On February 14, 2005, a suicide car bombing occurred in Beirut.
Former Lebanese Prime Minister Rafik Hariri and 14 others were killed.

Figure 3: Brief fully abstractive summaries on clusters D1001A-A, D1039G-A and D1043H-A, respectively on the Columbine massacre, the murder of Theo van Gogh and the assassination of Rafik Hariri.

combination of number and modifiers chosen must appear at least once as an IE rule match.

As for any generated text, a good summary also requires a text plan (Hovy, 1988) (McKeown, 1985). Ours consists of an ordering of the schemes. For example, an Attack summary begins with the scheme **event**. This ordering also determines which scheme to favor in the case of redundancy, e.g. given that a building was both damaged and destroyed, only the fact that is was destroyed will be mentioned.

4 Results and Discussion

We have implemented this fully abstractive summarization methodology. The abstraction schemes and text plan for the Attack category are written in an XML document, designed to easily allow the addition of more schemes and the design of new categories. The language processing of the source documents and the domain-specific knowledge are completely separate in the program.

Our system, which is meant as a proof of concept, can generate useful summaries for the Attack category, as can be seen in Figure 3. The key elements of information are present in each case, stated in a way that is easy to understand.

These short summaries have a high density of information, in terms of how much content from the source documents they cover for a given number of words. For example, using the most widely used content metric, Pyramid (Nenkova et al., 2007), the two sentences generated for the cluster D1001A-A contain 8 Semantic Content Units (SCU) for a weighted total of 30 out of a maximum of 56, for a raw Pyramid score of 0.54. Only 3 of the 43 automatic summaries beat this score on this cluster that year (the average was 0.31). Note that the summaries that we compare against contain up to 100 words, whereas ours is only 21 words long. We conclude that our method has the potential for creating summaries with much greater information density than the current state of the art.

In fact, our approach does not only have the potential to increase a summary's coverage, but also its linguistic quality and the reader satisfaction as well, since the most relevant information now appears at the beginning of the summary.

5 Conclusion and Future Work

We have developed and implemented a fully abstractive summarization methodology in the context of guided summarization. The higher density of information in our short summaries is one key to address the performance ceiling of extractive summarization methods. Although fully abstractive summarization is a daunting challenge, our work shows the feasibility and usefulness of this new direction for summarization research.

We are now expanding the variety and complexity of the abstraction schemes and generation patterns to deal with more aspects and other categories. We should then be able to compare on a greater scale the output of our system with the ones produced by other automatic systems and by humans on all the clusters used at TAC 2010 and 2011.

6 Acknowledgements

The authors want to thank Dr. Eduard Hovy, of ISI, and Prof. Kathy McKeown, of Columbia University, for fruitful discussions on abstractive summarization, and Dr. Judith Schlesinger and Dr. John Conroy, both of the IDA / Center for Computing Sciences, for providing us with their hand-crafted list of category- and aspect-relevant keywords.

References

R. Barzilay and L. Lee. 2004. Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization. *eprint arXiv:cs/0405039*, May.

Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *J. Artif. Int. Res.*, 34(1):637–674.

John M. Conroy, Judith D. Schlesinger, Peter A. Rankel, and Dianne P. O'Leary. 2010. CLASSY 2010: Summarization and metrics. In *Proceedings of the Third Text Analysis Conference*, Gaithersburg, Maryland, USA. National Institute of Standards and Technology.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, PA, USA.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the IEEE / ACL 2006 Workshop on Spoken Language Technology*. The Stanford Natural Language Processing Group.

Gerald DeJong, 1982. *An Overview of the FRUMP System*, pages 149–176. Lawrence Erlbaum.

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.

Albert Gatt and Ehud Reiter. 2009. SimpleNLG: a Realisation Engine for Practical Applications. In *ENLG '09: Proceedings of the 12th European Workshop on Natural Language Generation*, pages 90–93, Morristown, NJ, USA. Association for Computational Linguistics.

Pierre-Etienne Genest and Guy Lapalme. 2011. Framework for Abstractive Summarization using Text-to-Text Generation. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 64–73, Portland, Oregon, USA, June. Association for Computational Linguistics.

Pierre-Etienne Genest, Guy Lapalme, and Mehdi Yousfi-Monod. 2009. HexTac: the Creation of a Manual Extractive Run. In *Proceedings of the Second Text Analysis Conference*, Gaithersburg, Maryland, USA. National Institute of Standards and Technology.

Eduard H. Hovy. 1988. Planning coherent multisentential text. In *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, pages 163–169, Morristown, NJ, USA. Association for Computational Linguistics.

Heng Ji, Juan Liu, Benoit Favre, Dan Gillick, and Dilek Hakkani-Tur. 2010. Re-ranking summaries based on cross-document information extraction. In Pu-Jen Cheng, Min-Yen Kan, Wai Lam, and Preslav Nakov, editors, *Information Retrieval Technology*, volume 6458 of *Lecture Notes in Computer Science*, pages 432–442. Springer Berlin / Heidelberg. 10.1007/978-3-642-17187-1_42.

Karen Kipper, Anna Korhonen, Neville Ryant, and Martha Palmer. 2006. Extending VerbNet with Novel Verb Classes. In *LREC 2006*.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization - step one: Sentence compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 703–710. AAAI Press.

Kathleen R. McKeown. 1985. Discourse strategies for generating natural-language text. *Artif. Intell.*, 27:1–41, September.

Ani Nenkova, Rebecca Passonneau, and Kathleen McKeown. 2007. The pyramid method: Incorporating human content selection variation in summarization evaluation. *ACM Trans. Speech Lang. Process.*, 4, May.

Karolina Owczarzak and Hoa Trang Dang. 2011. Overview of the TAC 2011 summarization track: Guided task and aesop task. In *Proceedings of the Fourth Text Analysis Conference*, Gaithersburg, Maryland, USA. National Institute of Standards and Technology. http://www.nist.gov/tac/publications/.

Hideki Tanaka, Akinori Kinoshita, Takeshi Kobayakawa, Tadashi Kumano, and Naoto Kato. 2009. Syntax-driven sentence revision for broadcast news summarization. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation*, UCNLG+Sum '09, pages 39–47, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael White, Tanya Korelsky, Claire Cardie, Vincent Ng, David Pierce, and Kiri Wagstaff. 2001. Multidocument summarization via information extraction. In *Proceedings of the first international conference on Human language technology research*, HLT '01, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Assessing the Effect of Inconsistent Assessors on Summarization Evaluation

Karolina Owczarzak
National Institute of Standards and Technology
Gaithersburg, MD 20899
karolina.owczarzak@gmail.com

Peter A. Rankel
University of Maryland
College Park, Maryland
rankel@math.umd.edu

Hoa Trang Dang
National Institute of Standards and Technology
Gaithersburg, MD 20899
hoa.dang@nist.gov

John M. Conroy
IDA/Center for Computing Sciences
Bowie, Maryland
conroy@super.org

Abstract

We investigate the consistency of human assessors involved in summarization evaluation to understand its effect on system ranking and automatic evaluation techniques. Using Text Analysis Conference data, we measure annotator consistency based on human scoring of summaries for Responsiveness, Readability, and Pyramid scoring. We identify inconsistencies in the data and measure to what extent these inconsistencies affect the ranking of automatic summarization systems. Finally, we examine the stability of automatic metrics (ROUGE and CLASSY) with respect to the inconsistent assessments.

1 Introduction

Automatic summarization of documents is a research area that unfortunately depends on human feedback. Although attempts have been made at automating the evaluation of summaries, none is so good as to remove the need for human assessors. Human judgment of summaries, however, is not perfect either. We investigate two ways of measuring evaluation consistency in order to see what effect it has on summarization evaluation and training of automatic evaluation metrics.

2 Assessor consistency

In the Text Analysis Conference (TAC) Summarization track, participants are allowed to submit more than one run (usually two), and this option is often used to test different settings or versions of the same summarization system. In cases when the system versions are not too divergent, they sometimes produce identical summaries for a given topic. Summaries are randomized within each topic before they are evaluated, so the identical copies are usually interspersed with 40-50 other summaries for the same topic and are not evaluated in a row. Given that each topic is evaluated by a single assessor, it then becomes possible to check assessor consistency, i.e., whether the assessor judged the two identical summaries in the same way.

For each summary, assessors conduct content evaluation according to the Pyramid framework (Nenkova and Passonneau, 2004) and assign it Responsiveness and Readability scores[1], so assessor consistency can be checked in these three areas separately. We found between 230 (in 2009) and 430 (in 2011) pairs of identical summaries for the 2008-2011 data (given on average 45 topics, 50 runs, and two summarization conditions: main and update), giving in effect anywhere from around 30 to 60 instances per assessor per year. Using Krippendorff's *alpha* (Freelon, 2004), we calculated assessor consistency within each year, as well as total consistency over all years' data (for those assessors who worked multiple years). Table 1 shows rankings of assessors in 2011, based on their Readability, Responsiveness, and Pyramid judgments for identical summary pairs (around 60 pairs per assessor).

Interestingly, consistency values for Readability are lower overall than those for Responsiveness and Pyramid, even for the most consistent assessors. Given that Readability and Responsiveness are evaluated in the same way, i.e. by assigning a numerical score according to detailed guidelines, this sug-

[1] http://www.nist.gov/tac/2011/Summarization/GuidedSumm.2011.guidelines.html

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 359–362,
Jeju, Republic of Korea, 8-14 July 2012.

ID	Read	ID	Resp	ID	Pyr
G	0.867	G	0.931	G	0.975
D	0.866	D	0.875	D	0.970
A	0.801	H	0.808	H	0.935
H	0.783	A	0.750	A	0.931
F	0.647	F	0.720	E	0.909
C	0.641	E	0.711	C	0.886
E	0.519	C	0.490	F	0.872

Table 1: Annotator consistency in assigning Readability and Responsiveness scores and in Pyramid evaluation, as represented by Krippendorff's *alpha* for interval values, on 2011 data.

gests that Readability as a quality of text is inherently more vague and difficult to pinpoint.

On the other hand, Pyramid consistency values are generally the highest, which can be explained by how the Pyramid evaluation is designed. Even if the assessor is inconsistent in selecting Summary Content Units (SCUs) across different summaries, as long as the total summary weight is similar, the summary's final score will be similar, too.[2] Therefore, it would be better to look at whether assessors tend to find the same SCUs (information "nuggets") in different summaries on the same topic, and whether they annotate them consistently. This can be done using the "autoannotate" function of the Pyramid process, where all SCU contributors (selected text strings) from already annotated summaries are matched against the text of a candidate (un-annotated) summary. The autoannotate function works fairly well for matching between extractive summaries, which tend to repeat verbatim whole sentences from source documents.

For each summary in 2008-2011 data, we autoannotated it using all remaining manually-annotated summaries from the same topic, and then we compared the resulting "autoPyramid" score with the score from the original manual annotation for that summary. Ideally, the autoPyramid score should be lower or equal to the manual Pyramid score: it would mean that in this summary, the assessor selected as relevant all the same strings as s/he found in the other summaries on the same topic, plus possibly some more information that did not appear anywhere else. If the autoPyramid score is higher than the manual Pyramid score, it means that either (1) the assessor missed relevant strings in this summary, but found them in other summaries; or (2) the strings selected as relevant elsewhere in the topic were accidental, and as such not repeated in this summary. Either way, if we then average out score differences for all summaries for a given topic, it will give us a good picture of the annotation consistency in this particular topic. Higher average autoPyramid scores suggest that the assessor was missing content, or otherwise making frequent random mistakes in assigning content. Figure 1 shows the macro-average difference between manual Pyramid scores and autoPyramid scores for each assessor in 2011.[3] For the most part, it mirrors the consistency ranking from Table 1, confirming that some assessors are less consistent than others; however, certain differences appear: for instance, Assessor A is one of the most consistent in assigning Readability scores, but is not very good at selecting SCUs consistently. This can be explained by the fact that the Pyramid evaluation and assigning Readability scores are different processes and might require different skills and types of focus.

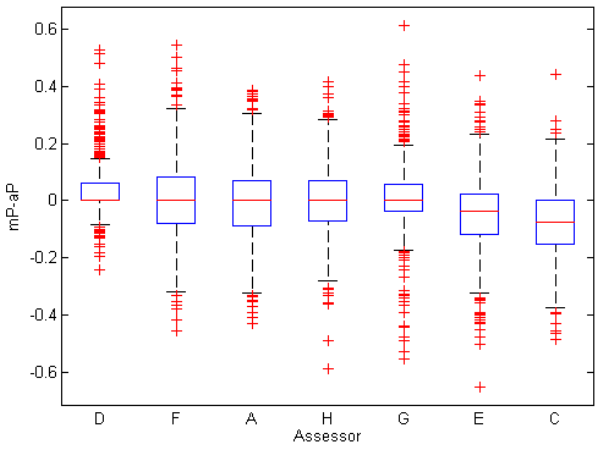


Figure 1: Annotator consistency in selecting SCUs in Pyramid evaluation, as represented by the difference between manual Pyramid and automatic Pyramid scores (mP-aP), on 2011 data.

3 Impact on evaluation

Since human assessment is used to rank participating summarizers in the TAC Summarization track,

[2]The final score is based on total weight of all SCUs found in the summary, so the same weight can be obtained by selecting a larger number of lower-weight SCUs or a smaller number of higher-weight SCUs (or the same number of similar-weight SCUs which nevertheless denote different content).

[3]Due to space constraints, we report figures for only 2011, but the results for other years are similar.

	Pearson's *r*		Spearman's *rho*	
	-1 worst	-2 worst	-1 worst	-2 worst
Readability	0.995	0.993	0.988	0.986
Responsiveness	0.996	0.989	0.986	0.946
Pyramid	0.996	0.992	0.978	0.960
mP-aP	0.996	0.987	0.975	0.943

Table 2: Correlation between the original summarizer ranking and the ranking after excluding topics by one or two worst assessors in each category.

we should examine the potential impact of inconsistent assessors on the overall evaluation. Because the final summarizer score is the average over many topics, and the topics are fairly evenly distributed among assessors for annotation, excluding noisy topics/assessors has very little impact on summarizer ranking. As an example, consider the 2011 assessor consistency data in Table 1 and Figure 1. If we exclude topics by the worst performing assessor from each of these categories, recalculate the summarizer rankings, and then check the correlation between the original and newly created rankings, we obtain results in Table 2.

Although the impact on evaluating automatic *summarizers* is small, it could be argued that excluding topics with inconsistent human scoring will have an impact on the performance of automatic *evaluation metrics*, which might be unfairly penalized by their inability to emulate random human mistakes. Table 3 shows ROUGE-2 (Lin, 2004), one of the state-of-the-art automatic metrics used in TAC, and its correlations with human metrics, before and after exclusion of noisy topics from 2011 data. The results are fairly inconclusive: it seems that in most cases, removing topics does more harm than good, suggesting that the signal-to-noise ratio is still tipped in favor of signal. The only exception is Readability, where ROUGE records a slight increase in correlation; this is unsurprising, given that consistency values for Readability are the lowest of all categories, and perhaps here removing noise has more impact. In the case of Pyramid, there is a small gain when we exclude the single worst assessor, but excluding two assessors results in a decreased correlation, perhaps because we remove too much valid information at the same time.

A different picture emerges when we examine how well ROUGE-2 can predict human scores on the *summary* level. We pooled together all summaries annotated by each particular assessor and calculated the correlation between ROUGE-2 and this assessor's manual scores for individual summaries. Then we calculated the mean correlation over all assessors. Unsurprisingly, inconsistent assessors tend to correlate poorly with automatic (and therefore always consistent) metrics, so excluding one or two worst assessors from each category increases ROUGE's average per-assessor summary-level correlation, as can be seen in Table 4. The only exception here is when we exclude assessors based on their autoPyramid performance: again, because inconsistent SCU selection doesn't necessarily translate into inconsistent final Pyramid scores, excluding those assessors doesn't do much for ROUGE-2.

	Readability	Responsiveness	Pyramid	mP-aP
before	0.705	0.930	0.954	0.954
-1 worst	0.718	0.921	0.961	0.942
-2 worst	0.718	0.904	0.952	0.923

Table 3: Correlation between the summarizer rankings according to ROUGE-2 and human metrics, before and after excluding topics by one or two worst assessors in that category.

	Readability	Responsiveness	Pyramid	mP-aP
before	0.579	0.694	0.771	0.771
-1 worst	0.626	0.695	0.828	0.752
-2 worst	0.628	0.721	0.817	0.741

Table 4: Correlation between ROUGE-2 and human metrics on a summary level before and after excluding topics by one or two worst assessors in that category.

4 Impact on training

Another area where excluding noisy topics might be useful is in training new automatic evaluation metrics. To examine this issue we turned to CLASSY (Rankel et al., 2011), an automatic evaluation metric submitted to TAC each year from 2009-2011. CLASSY consists of four different versions, each aimed at predicting a particular human evaluation score. Each version of CLASSY is based on one of three regression methods: robust regression, non-negative least squares, or canonical correlation. The regressions are calculated based on a collection of linguistic and content features, derived from the summary to be scored.

CLASSY requires two years of marked data to score summaries in a new year. In order to predict

the human metrics in 2011, for example, CLASSY uses the human ratings from 2009 and 2010. It first considers each subset of the features in turn, and using each of the regression methods, fits a model to the 2009 data. The subset/method combination that best predicts the 2010 scores is then used to predict scores for 2011. However, the model is first retrained on the 2010 data to calculate the coefficients to be used in predicting 2011.

First, we trained all four CLASSY versions on all available 2009-2010 topics, and then trained again excluding topics by the most inconsistent assessor(s). A different subset of topics was excluded depending on whether this particular version of CLASSY was aiming to predict Responsiveness, Readability, or the Pyramid score. Then we tested CLASSY's performance on 2011 data, ranking either automatic summarizers (NoModels case) or human and automatic summarizers together (AllPeers case), separately for main and update summaries, and calculated its correlation with the metrics it was aiming to predict. Table 5 shows the result of this comparison. For Pyramid, (a) indicates that excluded topics were selected based on Krippendorff's *alpha*, and (b) indicates that topics were excluded based on their mean difference between manual and automatic Pyramid scores.

The results are encouraging; it seems that removing noisy topics from training data does improve the correlations with manual metrics in most cases. The greatest increase takes place in CLASSY's correlations with Responsiveness for main summaries in AllPeers case, and for correlations with Readability. While none of the changes are large enough to achieve statistical significance, the pattern of improvement is fairly consistent.

	NoModels		AllPeers	
	main	update	main	update
Pyramid				
CLASSY1_Pyr	0.956	0.898	0.945	0.936
CLASSY1_Pyr_new (a)	0.950	0.895	0.932	**0.955**
CLASSY1_Pyr_new (b)	**0.960**	**0.900**	0.940	**0.955**
Responsiveness				
CLASSY2_Resp	0.951	0.903	0.948	0.963
CLASSY2_Resp_new	**0.954**	**0.907**	**0.973**	0.950
CLASSY4_Resp	0.951	0.927	0.830	0.949
CLASSY4_Resp_new	0.943	**0.928**	**0.887**	0.946
Readability				
CLASSY3_Read	0.768	0.705	0.844	0.907
CLASSY3_Read_new	**0.793**	**0.721**	**0.858**	0.906

Table 5: Correlations between CLASSY and human metrics on 2011 data (main and update summaries), before and after excluding most inconsistent topic from 2009-2010 training data for CLASSY.

5 Conclusions

We investigated the consistency of human assessors in the area of summarization evaluation. We considered two ways of measuring assessor consistency, depending on the metric, and studied the impact of consistent scoring on ranking summarization systems and on the performance of automatic evaluation systems. We found that summarization system ranking, based on scores for multiple topics, was surprisingly stable and didn't change significantly when several topics were removed from consideration. However, on a summary level, removing topics scored by the most inconsistent assessors helped ROUGE-2 increase its correlation with human metrics. In the area of training automatic metrics, we found some encouraging results; removing noise from the training data allowed most CLASSY versions to improve their correlations with the manual metrics that they were aiming to model.

References

Deen G. Freelon. 2010. ReCal: Intercoder Reliability Calculation as a Web Service. *International Journal of Internet Science*, Vol 5(1).

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, 78–81. Barcelona, Spain.

Ani Nenkova and Rebecca J. Passonneau. 2004. Evaluating content selection in summarization: The Pyramid method. *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 145–152. Boston, MA.

Rebecca J. Passonneau, Ani Nenkova, Kathleen McKeown, and Sergey Sigelman. 2005. Applying the Pyramid method in DUC 2005. *Proceedings of the 5th Document Understanding Conference (DUC)*. Vancouver, Canada.

Peter A. Rankel, John M. Conroy, and Judith D. Schlesinger. 2012. Better Metrics to Automatically Predict the Quality of a Text Summary. *Proceedings of the SIAM Data Mining Text Mining Workshop 2012*.

Fast and Robust Part-of-Speech Tagging Using Dynamic Model Selection

Jinho D. Choi
Department of Computer Science
University of Colorado Boulder
choijd@colorado.edu

Martha Palmer
Department of Linguistics
University of Colorado Boulder
mpalmer@colorado.edu

Abstract

This paper presents a novel way of improving POS tagging on heterogeneous data. First, two separate models are trained (generalized and domain-specific) from the same data set by controlling lexical items with different document frequencies. During decoding, one of the models is selected dynamically given the cosine similarity between each sentence and the training data. This dynamic model selection approach, coupled with a one-pass, left-to-right POS tagging algorithm, is evaluated on corpora from seven different genres. Even with this simple tagging algorithm, our system shows comparable results against other state-of-the-art systems, and gives higher accuracies when evaluated on a mixture of the data. Furthermore, our system is able to tag about 32K tokens per second. We believe that this model selection approach can be applied to more sophisticated tagging algorithms and improve their robustness even further.

1 Introduction

When it comes to POS tagging, two things must be checked. First, a POS tagger needs to be tested for its robustness in handling heterogeneous data.[1] Statistical POS taggers perform very well when their training and testing data are from the same source, achieving over 97% tagging accuracy (Toutanova et al., 2003; Giménez and Màrquez, 2004; Shen et al., 2007). However, the performance degrades increasingly as the discrepancy between the training and testing data gets larger. Thus, to ensure robustness, a tagger needs to be evaluated on several different kinds of data. Second, a POS tagger should be tested for its speed. POS tagging is often performed as a pre-processing step to other tasks (e.g., parsing, chunking) and it should not be a bottleneck for those tasks. Moreover, recent NLP tasks deal with very large-scale data where tagging speed is critical.

To improve robustness, we first train two separate models; one is optimized for a general domain and the other is optimized for a domain specific to the training data. During decoding, we dynamically select one of the models by measuring similarities between input sentences and the training data. Our hypothesis is that the domain-specific and generalized models perform better for sentences similar and not similar to the training data, respectively. In this paper, we describe how to build both models using the same training data and select an appropriate model given input sentences during decoding. Each model uses a one-pass, left-to-right POS tagging algorithm. Even with the simple tagging algorithm, our system gives results that are comparable to two other state-of-the-art systems when coupled with this dynamic model selection approach. Furthermore, our system shows noticeably faster tagging speed compared to the other two systems.

For our experiments, we use corpora from seven different genres (Weischedel et al., 2011; Nielsen et al., 2010). This allows us to check the performance of each system on different kinds of data when run individually or selectively. To the best of our knowledge, this is the first time that a POS tagger has been evaluated on such a wide variety of data in English.

[1] We use the term "heterogeneous data" as a mixture of data collected from several different sources.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 363–367,
Jeju, Republic of Korea, 8-14 July 2012.

2 Approach

2.1 Training generalized and domain-specific models using document frequency

Consider training data as a collection of documents where each document contains sentences focusing on a similar topic. For instance, in the Wall Street Journal corpus, a document can be an individual file or all files within each section.[2] To build a generalized model, lexical features (e.g., n-gram word-forms) that are too specific to individual documents should be avoided so that a classifier can place more weight on features common to all documents.

To filter out these document-specific features, a threshold is set for the document frequency of each lowercase simplified word-form (LSW) in the training data. A simplified word-form (SW) is derived by applying the following regular expressions sequentially to the original word-form, w. 'replaceAll' is a function that replaces all matches of the regular expression in w (the 1st parameter) with the specific string (the 2nd parameter). In a simplified word, all numerical expressions are replaced with 0.

1. w.replaceAll(`\d%`, 0) (e.g., 1% → 0)
2. w.replaceAll(`\$\d`, 0) (e.g., $1 → 0)
3. w.replaceAll(`^\.\d`, 0) (e.g., .1 → 0)
4. w.replaceAll(`\d(,|:|-|\/|\.)\d`, 0) (e.g., 1,2|1:2|1-2|1/2|1.2 → 0)
5. w.replaceAll(`\d+`, 0) (e.g., 1234 → 0)

A LSW is a decapitalized SW. Given a set of LSW's whose document frequencies are greater than a certain threshold, a model is trained by using only lexical features associated with these LSW's. For a generalized model, we use a threshold of 2, meaning that only lexical features whose LSW's occur in at least 3 documents of the training data are used. For a domain-specific model, we use a threshold of 1.

The generalized and domain-specific models are trained separately; their learning parameters are optimized by running n-fold cross-validation where n is the total number of documents in the training data and grid search on Liblinear parameters c and B (see Section 2.4 for more details about the parameters).

[2]For our experiments, we treat each section of the Wall Street Journal as one document.

2.2 Dynamic model selection during decoding

Once both generalized and domain-specific models are trained, alternative approaches can be adapted for decoding. One is to run both models and merge their outputs. This approach can produce output that is potentially more accurate than output from either model, but takes longer to decode because the merging cannot be processed until both models are finished. Instead, we take an alternative approach, that is to select one of the models dynamically given the input sentence. If the model selection is done efficiently, this approach runs as fast as running just one model, yet can give more robust performance.

The premise of this dynamic model selection is that the domain-specific model performs better for input sentences similar to its training space, whereas the generalized model performs better for ones that are dissimilar. To measure similarity, a set of SW's, say T, used for training the domain-specific model is collected. During decoding, a set of SW's in each sentence, say S, is collected. If the cosine similarity between T and S is greater than a certain threshold, the domain-specific model is selected for decoding; otherwise, the generalized model is selected.

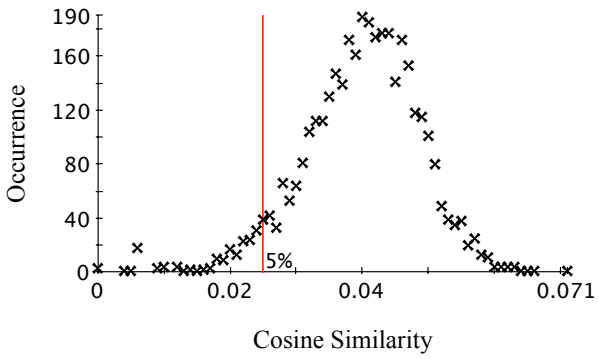


Figure 1: Cosine similarity distribution: the y-axis shows the number of occurrences for each cosine similarity during cross-validation.

The threshold is derived automatically by running cross-validation; for each fold, both models are run simultaneously and cosine similarities of sentences on which the domain-specific model performs better are extracted. Figure 1 shows the distribution of cosine similarities extracted during our cross-validation. Given the cosine similarity distribution, the similarity at the first 5% area (in this case, 0.025) is taken as the threshold.

2.3 Tagging algorithm and features

Each model uses a one-pass, left-to-right POS tagging algorithm. The motivation is to analyze how dynamic model selection works with a simple algorithm first and then apply it to more sophisticated ones later (e.g., bidirectional tagging algorithm).

Our feature set (Table 1) is inspired by Giménez and Màrquez (2004) although ambiguity classes are derived selectively for our case. Given a word-form, we count how often each POS tag is used with the form and keep only ones above a certain threshold. For both generalized and domain-specific models, a threshold of 0.7 is used, which keeps only POS tags used with their forms over 70% of the time. From our experiments, we find this to be more useful than expanding ambiguity classes with lower thresholds.

Lexical	$f_{i\pm\{0,1,2,3\}}$, $(m_{i-2,i-1})$, $(m_{i-1,i})$, $(m_{i-1,i+1})$, $(m_{i,i+1})$, $(m_{i+1,i+2})$, $(m_{i-2,i-1,i})$, $(m_{i-1,i,i+1})$, $(m_{i,i+1,i+2})$, $(m_{i-2,i-1,i+1})$, $(m_{i-1,i+1,i+2})$
POS	$p_{i-\{3,2,1\}}$, $a_{i+\{0,1,2,3\}}$, $(p_{i-2,i-1})$, $(a_{i+1,i+2})$, (p_{i-1}, a_{i+1}), (p_{i-2}, p_{i-1}, a_i), $(p_{i-2}, p_{i-1}, a_{i+1})$, (p_{i-1}, a_i, a_{i+1}), $(p_{i-1}, a_{i+1}, a_{i+2})$
Affix	$c_{:1}$, $c_{:2}$, $c_{:3}$, $c_{n:}$, $c_{n-1:}$, $c_{n-2:}$, $c_{n-3:}$
Binary	initial uppercase, all uppercase/lowercase, contains 1/2+ capital(s) not at the beginning, contains a (period/number/hyphen)

Table 1: Feature templates. i: the index of the current word, f: SW, m: LSW, p: POS, a: ambiguity class, c_*: character sequence in w_i (e.g., $c_{:2}$: the 1st and 2nd characters of w_i, $c_{n-1:}$: the n-1'th and n'th characters of w_i). See Giménez and Màrquez (2004) for more details.

2.4 Machine learning

Liblinear L2-regularization, L1-loss support vector classification is used for our experiments (Hsieh et al., 2008). From several rounds of cross-validation, learning parameters of ($c = 0.2$, $e = 0.1$, $B = 0.4$) and ($c = 0.1$, $e = 0.1$, $B = 0.9$) are found for the generalized and domain-specific models, respectively (c: cost, e: termination criterion, B: bias).

3 Related work

Toutanova et al. (2003) introduced a POS tagging algorithm using bidirectional dependency networks, and showed the best contemporary results. Giménez and Màrquez (2004) used one-pass, left-to-right and right-to-left combined tagging algorithm and achieved near state-of-the-art results. Shen et al. (2007) presented a tagging approach using guided learning for bidirectional sequence classification and showed current state-of-the-art results.[3]

Our individual models (generalized and domain-specific) are similar to Giménez and Màrquez (2004) in that we use a subset of their features and take one-pass, left-to-right tagging approach, which is a simpler version of theirs. However, we use Liblinear for learning, which trains much faster than their classifier, Support Vector Machines.

4 Experiments

4.1 Corpora

For training, sections 2-21 of the Wall Street Journal (WSJ) from OntoNotes v4.0 (Weischedel et al., 2011) are used. The entire training data consists of 30,060 sentences with 731,677 tokens. For evaluation, corpora from seven different genres are used: the MSNBC broadcasting conversation (BC), the CNN broadcasting news (BN), the Sinorama news magazine (MZ), the WSJ newswire (NW), and the GALE web-text (WB), all from OntoNotes v4.0. Additionally, the Mipacq clinical notes (CN) and the Medpedia articles (MD) are used for evaluation of medical domains (Nielsen et al., 2010). Table 2 shows distributions of these evaluation sets.

4.2 Accuracy comparisons

Our models are compared with two other state-of-the-art systems, the Stanford tagger (Toutanova et al., 2003) and the SVMTool (Giménez and Màrquez, 2004). Both systems are trained with the same training data and use configurations optimized for their best reported results. Tables 3 and 4 show tagging accuracies of all tokens and unknown tokens, respectively. Our individual models (Models D and G) give comparable results to the other systems. Model G performs better than Model D for BC, CN, and MD, which are very different from the WSJ. This implies that the generalized model shows its strength in tagging data that differs from the training data. The dynamic model selection approach (Model S) shows the most robust results across genres, although Models D and G still can perform

[3]Some semi-supervised and domain-adaptation approaches using external data had shown better performance (Daume III, 2007; Spoustová et al., 2009; Søgaard, 2011).

	BC	BN	CN	MD	MZ	NW	WB	Total
Source	MSNBC	CNN	Mipacq	Medpedia	Sinorama	WSJ	ENG	-
Sentences	2,076	1,969	3,170	1,850	1,409	1,640	1,738	13,852
All tokens	31,704	31,328	35,721	34,022	32,120	39,590	34,707	239,192
Unknown tokens	3,077	1,284	6,077	4,755	2,663	983	2,609	21,448

Table 2: Distributions of evaluation sets. The Total column indicates a mixture of data from all genres.

	BC	BN	CN	MD	MZ	NW	WB	Total
Model D	91.81	95.27	87.36	90.74	93.91	97.45	93.93	92.97
Model G	92.65	94.82	88.24	91.46	93.24	97.11	93.51	93.05
Model S	**92.26**	95.13	88.18	**91.34**	**93.88**	**97.46**	93.90	**93.21**
G over D	50.63	36.67	68.80	40.22	21.43	9.51	36.02	41.74
Stanford	87.71	**95.50**	**88.49**	90.86	92.80	97.42	**94.01**	92.50
SVMTool	87.82	95.13	87.86	90.54	92.94	97.31	93.99	92.32

Table 3: Tagging accuracies of all tokens (in %). Models D and G indicate domain-specific and generalized models, respectively and Model S indicates the dynamic model selection approach. "G over D" shows how often Model G is selected over Model D using the dynamic selection (in %).

	BC	BN	CN	MD	MZ	NW	WB	Total
Model S	**60.97**	77.73	68.69	**67.30**	**75.97**	**88.40**	76.27	**70.54**
Stanford	19.24	**87.31**	**71.20**	64.82	66.28	**88.40**	**78.15**	64.32
SVMTool	19.08	78.35	66.51	62.94	65.23	86.88	76.47	47.65

Table 4: Tagging accuracies of unknown tokens (in %).

better for individual genres (except for NW, where Model S performs better than any other model).

For both all and unknown token experiments, Model S performs better than the other systems when evaluated on a mixture of the data (the Total column). The differences are statistically significant for both experiments (McNemar's test, $p < .0001$). The Stanford tagger gives significantly better results for unknown tokens in BN; we suspect that this is where their bidirectional tagging algorithm has an advantage over our simple left-to-right algorithm.

4.3 Speed comparisons

Tagging speeds are measured by running each system on the mixture of all data. Our system and the Stanford system are both written in Java; the Stanford tagger provides APIs that allow us to make fair comparisons between the two systems. The SVMTool is written in Perl, so there is a systematic difference between the SVMTool and our system.

Table 5 shows speed comparisons between these systems. All experiments are evaluated on an Intel Xeon 2.57GHz machine. Our system tags about 32K tokens per second (0.03 milliseconds per token), which includes run-time for both POS tagging and model selection.

	Stanford	SVMTool	Model S
tokens / sec.	421	1,163	31,914

Table 5: Tagging speeds.

5 Conclusion

We present a dynamic model selection approach that improves the robustness of POS tagging on heterogeneous data. We believe that this approach can be applied to more sophisticated algorithms and improve their robustness even further. Our system also shows noticeably faster tagging speed against two other state-of-the-art systems. For future work, we will experiment with more diverse training and testing data and also more sophisticated algorithms.

Acknowledgments

This work was supported by the SHARP program funded by ONC: 90TR0002/01. The content is solely the responsibility of the authors and does not necessarily represent the official views of the ONC.

References

Hal Daume III. 2007. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL'07, pages 256–263.

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*, LREC'04.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. 2008. A Dual Coordinate Descent Method for Large-scale Linear SVM. In *Proceedings of the 25th international conference on Machine learning*, ICML'08, pages 408–415.

Rodney D. Nielsen, James Masanz, Philip Ogren, Wayne Ward, James H. Martin, Guergana Savova, and Martha Palmer. 2010. An architecture for complex clinical question answering. In *Proceedings of the 1st ACM International Health Informatics Symposium*, IHI'10, pages 395–399.

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*, ACL'07, pages 760–767.

Anders Søgaard. 2011. Semi-supervised condensed nearest neighbor for part-of-speech tagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, ACL'11, pages 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 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL'09, pages 763–771.

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 Annual Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, NAACL'03, pages 173–180.

Ralph Weischedel, Eduard Hovy, Martha Palmer, Mitch Marcus, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2011. OntoNotes: A Large Training Corpus for Enhanced Processing. In Joseph Olive, Caitlin Christianson, and John McCary, editors, *Handbook of Natural Language Processing and Machine Translation*. Springer.

Lemmatisation as a Tagging Task

Andrea Gesmundo
Department of Computer Science
University of Geneva
andrea.gesmundo@unige.ch

Tanja Samardžić
Department of Linguistics
University of Geneva
tanja.samardzic@unige.ch

Abstract

We present a novel approach to the task of word lemmatisation. We formalise lemmatisation as a category tagging task, by describing how a word-to-lemma transformation rule can be encoded in a single label and how a set of such labels can be inferred for a specific language. In this way, a lemmatisation system can be trained and tested using any supervised tagging model. In contrast to previous approaches, the proposed technique allows us to easily integrate relevant contextual information. We test our approach on eight languages reaching a new state-of-the-art level for the lemmatisation task.

1 Introduction

Lemmatisation and part-of-speech (POS) tagging are necessary steps in automatic processing of language corpora. This annotation is a prerequisite for developing systems for more sophisticated automatic processing such as information retrieval, as well as for using language corpora in linguistic research and in the humanities. Lemmatisation is especially important for processing morphologically rich languages, where the number of different word forms is too large to be included in the part-of-speech tag set. The work on morphologically rich languages suggests that using comprehensive morphological dictionaries is necessary for achieving good results (Hajič, 2000; Erjavec and Džeroski, 2004). However, such dictionaries are constructed manually and they cannot be expected to be developed quickly for many languages.

In this paper, we present a new general approach to the task of lemmatisation which can be used to overcome the shortage of comprehensive dictionaries for languages for which they have not been developed. Our approach is based on redefining the task of lemmatisation as a category tagging task. Formulating lemmatisation as a tagging task allows the use of advanced tagging techniques, and the efficient integration of contextual information. We show that this approach gives the highest accuracy known on eight European languages having different morphological complexity, including agglutinative (Hungarian, Estonian) and fusional (Slavic) languages.

2 Lemmatisation as a Tagging Task

Lemmatisation is the task of grouping together word forms that belong to the same inflectional morphological paradigm and assigning to each paradigm its corresponding canonical form called lemma. For example, English word forms *go*, *goes*, *going*, *went*, *gone* constitute a single morphological paradigm which is assigned the lemma *go*. Automatic lemmatisation requires defining a model that can determine the lemma for a given word form. Approaching it directly as a tagging task by considering the lemma itself as the tag to be assigned is clearly unfeasible: 1) the size of the tag set would be proportional to the vocabulary size, and 2) such a model would overfit the training corpus missing important morphological generalisations required to predict the lemma of unseen words (e.g. the fact that the transformation from *going* to *go* is governed by a general rule that applies to most English verbs).

Our method assigns to each word a label encod-

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 368–372,
Jeju, Republic of Korea, 8-14 July 2012.

ing the transformation required to obtain the lemma string from the given word string. The generic transformation from a word to a lemma is done in four steps: 1) remove a suffix of length N_s; 2) add a new lemma suffix, L_s; 3) remove a prefix of length N_p; 4) add a new lemma prefix, L_p. The tuple $\tau \equiv \langle N_s, L_s, N_p, L_p \rangle$ defines the word-to-lemma transformation. Each tuple is represented with a label that lists the 4 parameters. For example, the transformation of the word *going* into its lemma is encoded by the label $\langle 3, \emptyset, 0, \emptyset \rangle$. This label can be observed on a specific lemma-word pair in the training set but it generalizes well to the unseen words that are formed regularly by adding the suffix *-ing*. The same label applies to any other transformation which requires only removing the last 3 characters of the word string.

Suffix transformations are more frequent than prefix transformations (Jongejan and Dalianis, 2009). In some languages, such as English, it is sufficient to define only suffix transformations. In this case, all the labels will have N_p set to 0 and L_p set to $\emptyset$. However, languages richer in morphology often require encoding prefix transformations too. For example, in assigning the lemma to the negated verb forms in Czech the negation prefix needs to be removed. In this case, the label $\langle 1, t, 2, \emptyset \rangle$ maps the word *nevěděl* to the lemma *vědět*. The same label generalises to other (word, lemma) pairs: (*nedokázal*, *dokázat*), (*neexistoval*, *existovat*), (*nepamatoval*, *pamatovat*).[1]

The set of labels for a specific language is induced from a training set of pairs (word, lemma). For each pair, we first find the Longest Common Substring (LCS) (Gusfield, 1997). Then we set the value of N_p to the number of characters in the word that precede the start of LCS and N_s to the number of characters in the word that follow the end of LCS. The value of L_p is the substring preceding LCS in the lemma and the value of L_s is the substring following LCS in the lemma. In the case of the example pair (*nevěděl*, *vědět*), the LCS is *vědě*, 2 characters precede the LCS in the word and 1 follows it. There are no characters preceding the start of the LCS in the lemma and '*t*' follows it. The generated label is added to the set of labels.

[1]The transformation rules described in this section are well adapted for a wide range of languages which encode morphological information by means of affixes. Other encodings can be designed to handle other morphological types (such as Semitic languages).

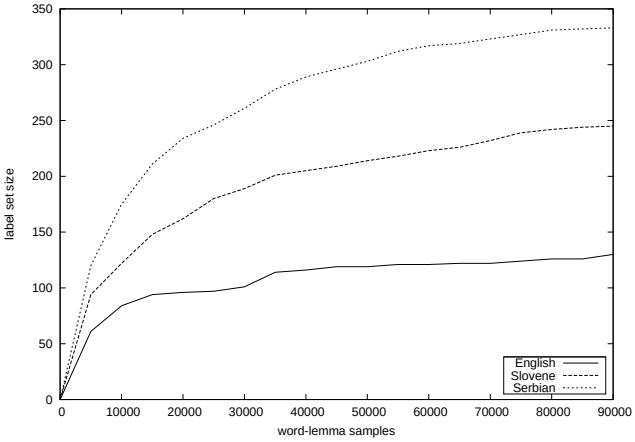


Figure 1: Growth of the label set with the number of training instances.

3 Label set induction

We apply the presented technique to induce the label set from annotated running text. This approach results in a set of labels whose size convergences quickly with the increase of training pairs.

Figure 1 shows the growth of the label set size with the number of tokens seen in the training set for three representative languages. This behavior is expected on the basis of the known interaction between the frequency and the regularity of word forms that is shared by all languages: infrequent words tend to be formed according to a regular pattern, while irregular word forms tend to occur in frequent words. The described procedure leverages this fact to induce a label set that covers most of the word occurrences in a text: a specialized label is learnt for frequent irregular words, while a generic label is learnt to handle words that follow a regular pattern.

We observe that the non-complete convergence of the label set size is, to a large extent, due to the presence of noise in the corpus (annotation errors, typos or inconsistency). We test the robustness of our method by deciding not to filter out the noise generated labels in the experimental evaluation. We also observe that encoding the prefix transformation in the label is fundamental for handling the size of the label sets in the languages that frequently use lemma prefixes. For example, the label set generated for

Czech doubles in size if only the suffix transformation is encoded in the label. Finally, we observe that the size of the set of induced labels depends on the morphological complexity of languages, as shown in Figure 1. The English set is smaller than the Slovene and Serbian sets.

4 Experimental Evaluation

The advantage of structuring the lemmatisation task as a tagging task is that it allows us to apply successful tagging techniques and use the context information in assigning transformation labels to the words in a text. For the experimental evaluations we use the Bidirectional Tagger with Guided Learning presented in Shen et al. (2007). We chose this model since it has been shown to be easily adaptable for solving a wide set of tagging and chunking tasks obtaining state-of-the-art performances with short execution time (Gesmundo, 2011). Furthermore, this model has consistently shown good generalisation behaviour reaching significantly higher accuracy in tagging unknown words than other systems.

We train and test the tagger on manually annotated G. Orwell's "1984" and its translations to seven European languages (see Table 2, column 1), included in the Multext-East corpora (Erjavec, 2010). The words in the corpus are annotated with both lemmas and detailed morphosyntactic descriptions including the POS labels. The corpus contains 6737 sentences (approximatively $110k$ tokens) for each language. We use 90% of the sentences for training and 10% for testing.

We compare lemmatisation performance in different settings. Each setting is defined by the set of features that are used for training and prediction. Table 1 reports the four feature sets used. Table 2 reports the accuracy scores achieved in each setting. We establish the Base Line (BL) setting and performance in the first experiment. This setting involves only features of the current word, $[w_0]$, such as the word form, suffixes and prefixes and features that flag the presence of special characters (digits, hyphen, caps). The BL accuracy is reported in the second column of Table 2).

In the second experiment, the BL feature set is expanded with features of the surrounding words ($[w_{-1}]$, $[w_1]$) and surrounding predicted lemmas ($[lem_{-1}]$, $[lem_1]$). The accuracy scores obtained in the second experiment are reported in the third column of Table 2. The consistent improvements over the BL scores for all the languages, varying from the lowest relative error reduction (RER) for Czech (5.8%) to the highest for Romanian (31.6%), confirm the significance of the context information. In the third experiment, we use a feature set in which the BL set is expanded with the predicted POS tag of the current word, $[pos_0]$.[2] The accuracy measured in the third experiment (Table 2, column 4) shows consistent improvement over the BL (the best RER is 34.2% for Romanian). Furthermore, we observe that the accuracy scores in the third experiment are close to those in the second experiment. This allows us to state that it is possible to design high quality lemmatisation systems which are independent of the POS tagging. Instead of using the POS information, which is currently standard practice for lemmatisation, the task can be performed in a context-wise setting using only the information about surrounding words and lemmas.

In the fourth experiment we use a feature set consisting of contextual features of words, predicted lemmas and predicted POS tags. This setting com-

Base Line (BL)	$[w_0]$, *flagChars*(w_0), *prefixes*(w_0), *suffixes*(w_0)
+ context	BL + $[w_1]$, $[w_{-1}]$, $[lem_1]$, $[lem_{-1}]$
+ POS	BL + $[pos_0]$
+cont.&POS	BL + $[w_1]$, $[w_{-1}]$, $[lem_1]$, $[lem_{-1}]$, $[pos_0]$, $[pos_{-1}]$, $[pos_1]$

Table 1: Feature sets.

	Base	+	+	+cont.&POS	
Language	Line	cont.	POS	Acc.	UWA
Czech	96.6	96.8	96.8	97.7	86.3
English	98.8	99.1	99.2	99.6	94.7
Estonian	95.8	96.2	96.5	97.4	78.5
Hungarian	96.5	96.9	97.0	97.5	85.8
Polish	95.3	95.6	96.0	96.8	85.8
Romanian	96.2	97.4	97.5	98.3	86.9
Serbian	95.0	95.3	96.2	97.2	84.9
Slovene	96.1	96.6	97.0	98.1	87.7

Table 2: Accuracy of the lemmatizer in the four settings.

[2]The POS tags that we use are extracted from the morphosyntactic descriptions provided in the corpus and learned using the same system that we use for lemmatisation.

bines the use of the context with the use of the predicted POS tags. The scores obtained in the fourth experiment are considerably higher than those in the previous experiments (Table 2, column 5). The RER computed against the BL varies between 28.1% for Hungarian and 66.7% for English. For this setting, we also report accuracies on unseen words only (UWA, column 6 in Table 2) to show the generalisation capacities of the lemmatizer. The UWA scores 85% or higher for all the languages except Estonian (78.5%).

The results of the fourth experiment show that interesting improvements in the performance are obtained by combining the POS and context information. This option has not been explored before. Current systems typically use only the information on the POS of the target word together with lemmatisation rules acquired separately from a dictionary, which roughly corresponds to the setting of our third experiment. The improvement in the fourth experiment compared to the third experiment (RER varying between 12.5% for Czech and 50% for English) shows the advantage of our context-sensitive approach over the currently used techniques.

All the scores reported in Table 2 represent performance with raw text as input. It is important to stress that the results are achieved using a general tagging system trained only a small manually annotated corpus, with no language specific external sources of data such as independent morphological dictionaries, which have been considered necessary for efficient processing of morphologically rich languages.

5 Related Work

Juršič et al. (2010) propose a general multilingual lemmatisation tool, LemGen, which is tested on the same corpora that we used in our evaluation. LemGen learns word transformations in the form of *ripple-down rules*. Disambiguition between multiple possible lemmas for a word form is based on the gold-standard morphosyntactic label of the word. Our system outperforms LemGen on all the languages. We measure a Relative Error Reduction varying between 81% for Serbian and 86% for English. It is worth noting that we do not use manually constructed dictionaries for training, while Juršič et al. (2010) use additional dictionaries for languages for which they are available.

Chrupała (2006) proposes a system which, like our system, learns the lemmatisation rules from a corpus, without external dictionaries. The mappings between word forms and lemmas are encoded by means of the *shortest edit script*. The sets of edit instructions are considered as class labels. They are learnt using a SVM classifier and the word context features. The most important limitation of this approach is that it cannot deal with both suffixes and prefixes at the same time, which is crucial for efficient processing of morphologically rich languages. Our approach enables encoding transformations on both sides of words. Furthermore, we propose a more straightforward and a more compact way of encoding the lemmatisation rules.

The majority of other methods are concentrated on lemmatising out-of-lexicon words. Toutanova and Cherry (2009) propose a joint model for assigning the set of possible lemmas and POS tags to out-of-lexicon words which is language independent. The lemmatizer component is a discriminative character transducer that uses a set of within-word features to learn the transformations from input data consisting of a lexicon with full morphological paradigms and unlabelled texts. They show that the joint model outperforms the pipeline model where the POS tag is used as input to the lemmatisation component.

6 Conclusion

We have shown that redefining the task of lemmatisation as a category tagging task and using an efficient tagger to perform it results in a performance that is at the state-of-the-art level. The adaptive general classification model used in our approach makes use of different sources of information that can be found in a small annotated corpus, with no need for comprehensive, manually constructed morphological dictionaries. For this reason, it can be expected to be easily portable across languages enabling good quality processing of languages with complex morphology and scarce resources.

7 Acknowledgements

The work described in this paper was partially funded by the Swiss National Science Foundation grants CRSI22 127510 (COMTIS) and 122643.

References

Grzegorz Chrupała. 2006. Simple data-driven context-sensitive lemmatization. In *Proceedings of the Sociedad Española para el Procesamiento del Lenguaje Natural*, volume 37, page 121131, Zaragoza, Spain.

Tomaž Erjavec and Sašo Džeroski. 2004. Machine learning of morphosyntactic structure: lemmatizing unknown Slovene words. *Applied Artificial Intelligence*, 18:17–41.

Tomaž Erjavec. 2010. Multext-east version 4: Multilingual morphosyntactic specifications, lexicons and corpora. 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 conference on International Language Resources and Evaluation (LREC'10)*, pages 2544–2547, Valletta, Malta. European Language Resources Association (ELRA).

Andrea Gesmundo. 2011. Bidirectional sequence classification for tagging tasks with guided learning. In *Proceedings of TALN 2011*, Montpellier, France.

Dan Gusfield. 1997. *Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology*. Cambridge University Press.

Jan Hajič. 2000. Morphological tagging: data vs. dictionaries. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 94–101, Seattle, Washington. Association for Computational Linguistics.

Bart Jongejan and Hercules Dalianis. 2009. Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. 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 145–153, Suntec, Singapore, August. Association for Computational Linguistics.

Matjaž Juršič, Igor Mozetič, Tomaž Erjavec, and Nada Lavrač. 2010. LemmaGen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science*, 16(9):1190–1214.

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, Prague, Czech Republic. Association for Computational Linguistics.

Kristina Toutanova and Colin Cherry. 2009. A global model for joint lemmatization and part-of-speech prediction. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, page 486494, Suntec, Singapore. Association for Computational Linguistics.

How Are Spelling Errors Generated and Corrected? A Study of Corrected and Uncorrected Spelling Errors Using Keystroke Logs

Yukino Baba
The University of Tokyo
yukino.baba@gmail.com

Hisami Suzuki
Microsoft Research
hisamis@microsoft.com

Abstract

This paper presents a comparative study of spelling errors that are corrected as you type, vs. those that remain uncorrected. First, we generate naturally occurring online error correction data by logging users' keystrokes, and by automatically deriving pre- and post-correction strings from them. We then perform an analysis of this data against the errors that remain in the final text as well as across languages. Our analysis shows a clear distinction between the types of errors that are generated and those that remain uncorrected, as well as across languages.

1 Introduction

When we type text using a keyboard, we generate many spelling errors, both typographical (caused by the keyboard layout and hand/finger movement) and cognitive (caused by phonetic or orthographic similarity) (Kukich, 1992). When the errors are caught during typing, they are corrected on the fly, but unnoticed errors will persist in the final text. Previous research on spelling correction has focused on the latter type (which we call **uncorrected errors**), presumably because the errors that are corrected on the spot (referred to here as **corrected errors**) are not recoded in the form of a text. However, studying corrected errors is important for at least three reasons. First, such data encapsulates the spelling mistake and correction by the author, in contrast to the case of uncorrected errors in which the intended correction is typically assigned by a third person (an annotator), or by an automatic method (Whitelaw et al., 2009; Aramaki et al., 2010)[1]. Secondly, data on corrected errors will enable us to build a spelling correction application that targets correction on the fly, which directly reduces the number of keystrokes in typing. This is crucial for languages that use transliteration-based text input methods, such as Chinese and Japanese, where a spelling error in the input Roman keystroke sequence will prevent the correct candidate words from appearing in the list of candidates in their native scripts, thereby preventing them from being entered altogether. Finally, we can collect a large amount of spelling errors and their corrections by logging keystrokes and extracting the pre- and post-correction strings from them. By learning the characteristics of corrected and uncorrected errors, we can expect to use the data for improving the correction of the errors that persisted in the final text as well.

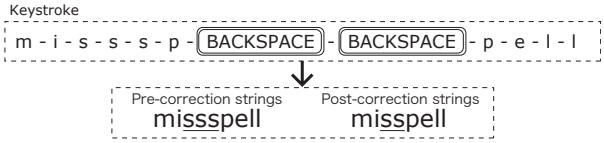


Figure 1: Example of keystroke

In this paper, we collect naturally occurring spelling error data that are corrected by the users online from keystroke logs, through the crowdsourcing infrastructure of Amazon's Mechanical Turk (MTurk). As detailed in Section 3, we display images to the worker of MTurk, and collect the descriptions of these images, while logging their keystrokes including the usage of backspace keys, via a crowd-based text input service. We collected logs for two typologically different languages, English and Japanese. An example of a log along with the extracted pre- and post-correction strings is shown in Figure 1. We then performed two comparative analyses: corrected vs. uncorrected errors in English (Section 4.3), and English vs. Japanese corrected errors (Section 4.4). Finally, we remark on an additional cause of spelling errors observed in all the data we analyzed (Section 4.5).

2 Related Work

Studies on spelling error generation mechanisms are found in earlier work such as Cooper (1983). In particular, Grudin (1983) offers a detailed study of the errors generated in the transcription typing scenario, where the subjects are asked to transcribe a text without correcting the errors they make. In a more recent work, Aramaki et al. (2010) automatically extracted error-correction candidate pairs from Twitter data based on the assumption that these pairs

[1] Using web search query logs is one notable exception, which only targets spelling errors in search queries (Gao et al., 2010)

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 373–377,
Jeju, Republic of Korea, 8-14 July 2012.

fall within a small edit distance, and that the errors are not in the dictionary and substantially less frequent than the correctly spelled counterpart. They then studied the effect of five factors that cause errors by building a classifier that uses the features associated with these classes and running ablation experiments. They claim that finger movements cause the spelling errors to be generated, but the uncorrected errors are characterized by visual factors such as the visual similarity of confused letters. Their experiments however target only the persisted errors, and their claim is not based on the comparison of generated and persisted errors.

Outside of English, Zheng et al. (2011) analyzed the keystroke log of a commercial text input system for Simplified Chinese, and compared the error patterns in Chinese with those in English. Their use of the keystroke log is different from ours in that they did not directly log the input in pinyin (Romanized Chinese by which native characters are input), but the input pinyin sequences are recovered from the Chinese words in the native script (hanzi) after the character conversion has already applied.

3 Keystroke Data Collection

Amazon's Mechanical Turk (MTurk) is a web service that enables crowdsourcing of tasks that are difficult for computers to solve, and has become an important infrastructure for gathering data and annotation for NLP research in recent years (Snow et al. 2008). To the extent of our knowledge, our work is the first to use this infrastructure to gather user keystroke data.

3.1 Task design

In order to collect naturally occurring keystrokes, we have designed two types of tasks, both of which consist of writing something about images. In one task type, we asked the workers to write a short description of images (image description task); in the other, the workers were presented with images of a person or an animal, and were asked to guess and type what she/he was saying (let-them-talk task). Using images as triggers for typing keeps the underlying motivation of keystroke collection hidden from the workers, simultaneously allowing language-independent data collection. For the image triggers, we used photos from the Flickr's Your Best Shot 2009/2010 groups . Examples of the tasks and collected text are given in Figure 2.

Figure 2: Examples of tasks and collected text (Translated text: "A flock of penguines are marching in the snow." and "Mummy, my feet can't touch the bottom.")

3.2 Task interface

For logging the keystrokes including the use of backspaces, we designed an original interface for the text boxes in the MTurk task. In order to simplify the interpretation of the log, we disabled the cursor movements and text highlighting via a mouse or the arrow keys in the text box; the workers are therefore forced to use the backspace key to make corrections. In Japanese, many commercially available text input methods (IMEs) have an auto-complete feature which prevents us from collecting all keystrokes for inputting a word. We therefore used an in-house IME that has disabled this feature to collect logs. This IME is hosted as a web service, and keystroke logs are also collected through the service. For English, we used the service for log collection only.

4 Keystroke Log Analysis

4.1 Data

We used both keystroke-derived and previously available error data for our analysis.

Keystroke-derived error pairs for English and Japanese (en_keystroke, ja_keystroke): from the raw keystroke logs collected using the method described in Section 3, we extracted only those words that included a use of the backspace key. We then recovered the strings before and after correction by the following steps (Cf. Figure 1):

- To recover the post-correction string, we deleted the same number of characters preceding a sequence of backspace keys.
- To recover the pre-correction string, we compared the prefix of the backspace usage (`misssp` in Figure 1) with the substrings after error correction (`miss`, `missp`, ⋯, `misspell`), and considered that the prefix was spell-corrected into the substring which is the longest and with the smallest edit distance

(in this case, `misssp` is an error for `missp`, so the pre-correction string is `missspell`).

We then lower-cased the pairs and extracted only those within the edit distance of 2. The resulting data which we used for our analysis consists of 44,104 pairs in English and 4,808 pairs in Japanese[2].

Common English errors (en_common): following previous work (Zheng et al., 2011), we obtained word pairs from Wikipedia[3] and SpellGood[4]. We lower-cased the entries from these sources, removed the duplicates and the pairs that included non-Roman alphabet characters, and extracted only those pairs within the edit distance of 2. This left us with 10,608 pairs.

4.2 Factors that affect errors

Spelling errors have traditionally been classified into four descriptive types: Deletion, Insertion, Substitution and Transposition (Damerau, 1964). For each of these types, we investigated the potential causes of error generation and correction, following previous work (Aramaki et al., 2010; Zheng et al., 2011). Physical factors: (1) motor control of hands and fingers; (2) distance between the keys; Visual factors: (3) visual similarity of characters; (4) position in a word; (5) same character repetition; Phonological factors: (6) phonological similarity of characters/words.

In what follows, our discussion is based on the frequency ratio of particular error types, where the frequency ratio refers to the number of cases in spelling errors divided by the total number of cases in all data. For example, the frequency ratio of consonant deletion is calculated by dividing the number of missing consonants in errors by the total number of consonants.

4.3 Corrected vs. uncorrected errors in English

In this subsection, we compare corrected and uncorrected errors of English, trying to uncover what factors facilitate the error correction.

Error types (Figure 3) Errors in en_keystroke are dominated by Substitution, while Deletion errors are the most common in en_common, indicating that

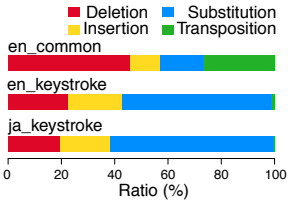


Figure 3: Ratios of error types

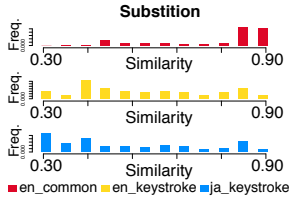


Figure 4: Visual similarities of characters in substitution errors

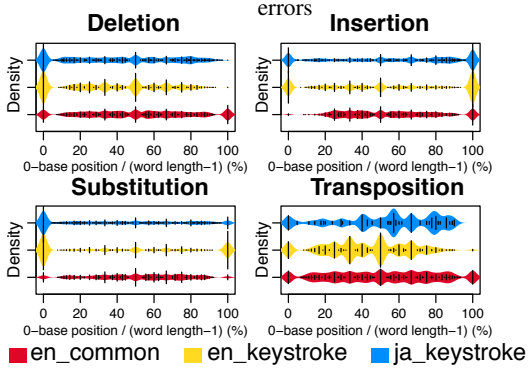


Figure 5: Positions of errors within words

Substitution mistakes are easy to catch, while Deletion mistakes tend to escape our attention. Zheng et al. (2011) reports that their pinyin correction errors are dominated by Deletion, which suggests that their log does in fact reflect the characteristics of corrected errors.

Position of error within a word (Figure 5) In en_keystroke, Deletion errors at the word-initial position are the most common, while Insertion and Substitution errors tend to occur both at the beginning and the end of a word. In contrast, in en_common, all error types are more prone to occur word-medially. This means that errors at word edges are corrected more often than the word-internal errors, which can be attributed to cognitive effect known as the bathtub effect (Aitchison, 1994), which states that we memorize words at the periphery most effectively in English.

Effect of character repetition (Figure 6) Deletion errors where characters are repeated, as in `tomorow`→`tomorrow`, is observed significantly more frequently than in a non-repeating context in en_common, but no such difference is observed in en_keystroke, showing that visually conspicuous errors tend to be corrected.

Visual similarity in Substitution errors (Figure 4) We computed the visual similarity of characters by $2\times(the\ area\ of\ overlap\ between\ character\ A\ and\ B)/(area\ of\ character\ A+area\ of\ character\ B)$ follow-

[2]The data is available for research purposes under `http://research.microsoft.com/research/downloads/details/4eb8d4a0-9c4e-4891-8846-7437d9dbd869/details.aspx`

[3]`http://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings/For_machines`

[4]`http://www.spellgood.net/sitemap.html`

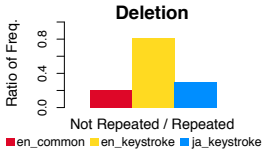


Figure 6: Effect of character repetition in Deletion

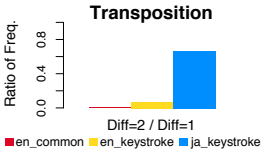


Figure 7: Difference of positions within words in Transposition

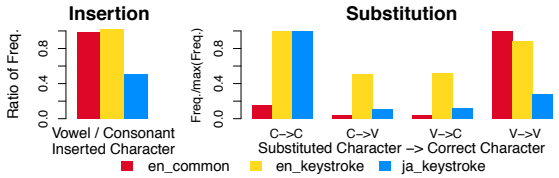


Figure 8: Consonants/vowels in Insertion and Substitution

ing Aramaki et al. (2010)[5]. Figure 4 shows that in en_common, Substitution errors of visually similar characters (e.g., `yoqa→yoga`) are in fact very common, while in en_keystroke, no such tendency is observed.

Phonological similarity in Substitution errors (Figure 8) In en_keystroke, there is no notable difference in consonant-to-consonant (C→C) and vowel-to-vowel (V→V) errors, but in en_common, V→V errors are overwhelmingly more common, suggesting that C→C can easily be noticed (e.g., `eazy→easy`) while V→V errors (e.g., `visable→visible`) are not. This tendency is consistent with the previous work on the cognitive distinction between consonants and vowels in English: consonants carry more lexical information than vowels (Nespor et al., 2003), a claim also supported by distributional evidence (Tanaka-Ishii, 2008). It may also be attributed to the fact that English vowel quality is not always reflected by the onthography in the straightforward maner.

Summarizing, we have observed both visual and phonological factors affect the correction of errors. Aramaki et al. (2010)'s experiments did not show that C/V distinction affect the errors, while our data shows that it does in the correction of errors.

4.4 Errors in English vs. Japanese

From Figure 3, we can see that the general error pattern is very similar between en_keystroke and ja_keystroke. Looking into the details, we discovered some characteristic errors in Japanese, which are phonologically and orthographically motivated.

Syllable-based transposition errors (Figure 7) When comparing the transposition errors by their distance, 1 being a transposition of adjacent characters and 2 a transposition skipping a character, the instances in en_keystroke are mostly of distance of 1, while in ja_keystroke, the distance of 2 also occurs commonly (e.g., `kotoro→tokoro`). This is interesting, because the Japanese writing system called kana is a syllabary system, and our data suggests that users may be typing a kana character (typically CV) as a unit. Furthermore, 73% of these errors share the vowel of the transposed syllables, which may be serving as a strong condition for this type of error.

[5]We calculated the area using the Courier New font which we used in our task interface.

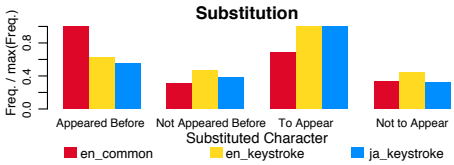


Figure 9: Look-ahead and Look-behind in Substitution

Errors in consonants/vowels (Figure 8) Errors in ja_keystroke are characterized by a smaller ratio of insertion errors of vowels relative to consonants, and by a relatively smaller ratio of V→V substitution errors. Both point to the relative robustness of inputting vowels as opposed to consonants in Japanese. Unlike English, Japanese only has five vowels whose pronunciations are transparently carried by the orthography; they are therefore expected to be less prone to cognitive errors.

4.5 Look-ahead and look-behind errors

In Substitution errors for all data we analyzed, substituting for the character that appeared before, or are to appear in the word was common (Figure 9). In particular, in en_keystroke and ja_keystroke, look-ahead errors are much more common than non-look-ahead errors. Grudin (1983) reports cases of permutation (e.g., `gib→big`) but our data includes non-permutation look-ahead errors such as `puclic→public` and `otigaga→otibaga`.

5 Conclusion

We have presented our collection methodology and analysis of error correction logs across error types (corrected vs. uncorrected) and languages (English and Japanese). Our next step is to utilize the collected data and analysis results to build online and offline spelling correction models.

Acknowledgments

This work was conducted during the internship of the first author at Microsoft Research. We are grateful to the colleagues for their help and feedback in conducting this research.

References

Aitchison, J. 1994. *Words in the Mind*. Blackwell.

Aramaki, E., R. Uno and M. Oka. 2010. TYPO Writer: ヒトはどのように打ち間違えるのか？ (TYPO Writer: how do humans make typos?). In *Proceedings of the 16th Annual Meeting of the Natural Language Society (in Japanese)*.

Cooper, W. E. (ed.) 1983. *Cognitive Aspects of Skilled Typewriting*. Springer-Verlag.

Damerau, F. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM 7(3): 659-664*.

Gao, J., X. Li, D. Micol, C. Quirk and X. Sun. 2010. A large scale ranker-based system for search query spelling correction. In *Proceedings of COLING*.

Grudin, J. T. 1983. Error patterns in novice and skilled transcription typing. In Cooper, W.E. (ed.), *Cognitive Aspects of Skilled Typewriting*. Springer-Verlag.

Kukich, K. 1992. Techniques for automatically correcting words in text. In *ACM Computing Surveys, 24(4)*.

Nespor, M., M. Peña, and J. Mehler. 2003. On the different roles of vowels and consonants in speech processing and language acquisition. *Lingue e Linguaggio*, pp. 221—247.

Snow, R., B. O'Connor, D. Jurafsky, and A. Ng. 2008. Cheap and fast – but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of EMNLP*.

Tanaka-Ishii, K. 2008. 単語に内在する情報量の偏在 (On the uneven distribution of information in words). In *Proceedings of the 14th Annual Meeting of the Natural Language Society (in Japanese)*.

Whitelaw, Casey, Ben Hutchinson, Grace Y. Chung, and Gerard Ellis. 2009. Using the web for language independent spellchecking and autocorrection. In *Proceedings of ACL*.

Zheng, Y., L. Xie, Z. Liu, M. Sun. Y. Zhang and L. Ru 2011. Why press backspace? Understanding user input behaviors in Chinese pinyin input method. In *Proceedings of ACL*

Tokenization: Returning to a Long Solved Problem
A Survey, Contrastive Experiment, Recommendations, and Toolkit

Rebecca Dridan & Stephan Oepen
Institutt for Informatikk, Universitetet i Oslo
{ rdridan|oe } @ifi.uio.no

Abstract

We examine some of the frequently disregarded subtleties of tokenization in Penn Treebank style, and present a new rule-based preprocessing toolkit that not only reproduces the Treebank tokenization with unmatched accuracy, but also maintains exact stand-off pointers to the original text and allows flexible configuration to diverse use cases (e.g. to genre- or domain-specific idiosyncrasies).

1 Introduction—Motivation

The task of *tokenization* is hardly counted among the grand challenges of NLP and is conventionally interpreted as breaking up "natural language text [...] into distinct meaningful units (or tokens)" (Kaplan, 2005). Practically speaking, however, tokenization is often combined with other string-level preprocessing—for example normalization of punctuation (of different conventions for dashes, say), disambiguation of quotation marks (into opening vs. closing quotes), or removal of unwanted mark-up—where the specifics of such pre-processing depend both on properties of the input text as well as on assumptions made in downstream processing.

Applying some string-level normalization *prior* to the identification of token boundaries can improve (or simplify) tokenization, and a sub-task like the disambiguation of quote marks would in fact be hard to perform *after* tokenization, seeing that it depends on adjacency to whitespace. In the following, we thus assume a *generalized* notion of tokenization, comprising all string-level processing up to and including the conversion of a sequence of characters (a string) to a sequence of token objects.[1]

Arguably, even in an overtly 'separating' language like English, there can be token-level ambiguities that ultimately can only be resolved through parsing (see § 3 for candidate examples), and indeed Waldron et al. (2006) entertain the idea of downstream processing on a token *lattice*. In this article, however, we accept the tokenization conventions and sequential nature of the Penn Treebank (PTB; Marcus et al., 1993) as a useful point of reference—primarily for interoperability of different NLP tools.

Still, we argue, there is remaining work to be done on PTB-compliant tokenization (reviewed in§ 2), both methodologically, practically, and technologically. In § 3 we observe that state-of-the-art tools perform poorly on re-creating PTB tokenization, and move on in § 4 to develop a modular, parameterizable, and transparent framework for tokenization. Besides improvements in tokenization accuracy and adaptability to diverse use cases, in § 5 we further argue that each token object should unambiguously link back to an underlying element of the original input, which in the case of tokenization of text we realize through a notion of *characterization*.

2 Common Conventions

Due to the popularity of the PTB, its tokenization has been a de-facto standard for two decades. Approximately, this means splitting off punctuation into separate tokens, disambiguating straight quotes, and separating contractions such as *can't* into *ca* and *n't*. There are, however, many special cases—

[1] Obviously, some of the normalization we include in the tokenization task (in this generalized interpretation) could be left to downstream analysis, where a tagger or parser, for example, could be expected to accept non-disambiguated quote marks (so-called straight or typewriter quotes) and disambiguate as part of syntactic analysis. However, on the (predominant) point of view that punctuation marks form tokens in their own right, the tokenizer would then have to adorn quote marks in some way, as to whether they were split off the left or right periphery of a larger token, to avoid unwanted syntactic ambiguity. Further, increasing use of Unicode makes texts containing 'natively' disambiguated quotes more common, where it would seem unfortunate to discard linguistically pertinent information by normalizing towards the poverty of pure ASCII punctuation.

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 378–382,
Jeju, Republic of Korea, 8-14 July 2012.

documented and undocumented. In much tagging and parsing work, PTB data has been used with gold-standard tokens, to a point where many researchers are unaware of the existence of the original 'raw' (untokenized) text. Accordingly, the formal definition of PTB tokenization[2] has received little attention, but reproducing PTB tokenization automatically actually is not a trivial task (see § 3).

As the NLP community has moved to process data other than the PTB, some of the limitations of the PTB tokenization have been recognized, and many recently released data sets are accompanied by a note on tokenization along the lines of: *Tokenization is similar to that used in PTB, except ...* Most exceptions are to do with hyphenation, or special forms of named entities such as chemical names or URLs. None of the documentation with extant data sets is sufficient to fully reproduce the tokenization.[3]

The CoNLL 2008 Shared Task data actually provided two forms of tokenization: that from the PTB (which many pre-processing tools would have been trained on), and another form that splits (most) hyphenated terms. This latter convention recently seems to be gaining ground in data sets like the Google 1T n-gram corpus (LDC#2006T13) and OntoNotes (Hovy et al., 2006). Clearly, as one moves towards a more application- and domain-driven idea of 'correct' tokenization, a more transparent, flexible, and adaptable approach to string-level pre-processing is called for.

3 A Contrastive Experiment

To get an overview of current tokenization methods, we recovered and tokenized the raw text which was the source of the (Wall Street Journal portion of the) PTB, and compared it to the gold tokenization in the syntactic annotation in the treebank.[4] We used three common methods of tokenization: (a) the original PTB tokenizer.sed script; (b) the tokenizer from the Stanford CoreNLP tools[5]; and (c) tokenization from the parser of Charniak & Johnson (2005). Table 1 shows quantitative differences between each of the three methods and the PTB, both in terms of the number of sentences where the tokenization differs, and also in the total Levenshtein distance (Levenshtein, 1966) over tokens (for a total of 49,208 sentences and 1,173,750 gold-standard tokens).

Tokenization Method	Differing Sentences	Levenshtein Distance
tokenizer.sed	3264	11168
CoreNLP	1781	3717
C&J parser	2597	4516

Table 1: Quantitative view on tokenization differences.

Looking at the differences qualitatively, the most consistent issue across all tokenization methods was ambiguity of sentence-final periods. In the treebank, final periods are always (with about 10 exceptions) a separate token. If the sentence ends in *U.S.* (but not other abbreviations, oddly), an extra period is hallucinated, so the abbreviation also has one. In contrast, C&J add a period to all final abbreviations, CoreNLP groups the final period with a final abbreviation and hence lacks a sentence-final period token, and the sed script strips the period off *U.S.* The 'correct' choice in this case is not obvious and will depend on how the tokens are to be used.

The majority of the discrepancies in the sed script tokenization come from an under-restricted punctuation rule that incorrectly splits on commas within numbers or ampersands within names. Other than that, the problematic cases are mostly shared across tokenization methods, and include issues with currencies, Irish names, hyphenization, and quote disambiguation. In addition, C&J make some additional modifications to the text, lemmatising expressions such as *won't* as *will* and *n't*.

4 REPP: A Generalized Framework

For tokenization to be studied as a first-class problem, and to enable customization and flexibility to diverse use cases, we suggest a non-procedural, rule-based framework dubbed REPP (Regular

[2] See `http://www.cis.upenn.edu/~treebank/tokenization.html` for available 'documentation' and a sed script for PTB-style tokenization.

[3] Øvrelid et al. (2010) observe that tokenizing with the GENIA tagger yields mismatches in one of five sentences of the GENIA Treebank, although the GENIA guidelines refer to scripts that may be available on request (Tateisi & Tsujii, 2006).

[4] The original WSJ text was last included with the 1995 release of the PTB (LDC#95T07) and required alignment with the treebank, with some manual correction so that the same text is represented in both raw and parsed formats.

[5] See `http://nlp.stanford.edu/software/corenlp.shtml`, run in '`strictTreebank3`' mode.

```
>wiki
#1
!([^␣])([])}?!,;:"'])␣([^␣]|$)    \1␣\2␣\3
!(^|[^␣])␣([[({“‘])([^␣])         \1␣\2␣\3
#
>1
:[[:space:]]+
```

Figure 1: Simplified examples of tokenization rules.

Expression-Based Pre-Processing)—essentially a cascade of ordered finite-state string rewriting rules, though transcending the formal complexity of regular languages by inclusion of (a) full perl-compatible regular expressions and (b) fixpoint iteration over groups of rules. In this approach, a first phase of string-level substitutions inserts whitespace around, for example, punctuation marks; upon completion of string rewriting, token boundaries are stipulated between all whitespace-separated substrings (and only these).

For a good balance of human and machine readability, REPP tokenization rules are specified in a simple, line-oriented textual form. Figure 1 shows a (simplified) excerpt from our PTB-style tokenizer, where the first character on each line is one of four REPP operators, as follows: (a) '`#`' for group formation; (b) '`>`' for group invocation, (c) '`!`' for substitution (allowing capture groups), and (d) '`:`' for token boundary detection.[6] In Figure 1, the two rules stripping off prefix and suffix punctuation marks adjacent to whitespace (i.e. matching the tab-separated left-hand side of the rule, to replace the match with its right-hand side) form a numbered group ('`#1`'), which will be iterated when called ('`>1`') until none of the rules in the group fires (a fixpoint). In this example, conditioning on whitespace adjacency avoids the issues observed with the PTB sed script (e.g. token boundaries within comma-separated numbers) and also protects against infinite loops in the group.[7]

REPP rule sets can be organized as modules, typically each in a file of its own, and invoked selectively by name (e.g. '`>wiki`' in Figure 1); to date, there exist modules for quote disambiguation, (relevant subsets of) various mark-up languages (HTML, LaTeX, wiki, and XML), and a handful of robustness rules (e.g. seeking to identify and repair 'sandwiched' inter-token punctuation). Individual tokenizers are configured at run-time, by selectively activating a set of modules (through command-line options). An open-source reference implementation of the REPP framework (in C++) is available, together with a library of modules for English.

5 Characterization for Traceability

Tokenization, and specifically our notion of generalized tokenization which allows text normalization, involves changes to the original text being analyzed, rather than just additional annotation. As such, full *traceability* from the token objects to the original text is required, which we formalize as 'characterization', in terms of character position links back to the source.[8] This has the practical benefit of allowing downstream analysis as direct (stand-off) annotation on the source text, as seen for example in the ACL Anthology Searchbench (Schäfer et al., 2011).

With our general regular expression replacement rules in REPP, making precise what it means for a token to link back to its 'underlying' substring requires some care in the design and implementation. Definite characterization links between the string before ($\mathcal{I}$) and after ($\mathcal{O}$) the application of a single rule can only be established in certain positions, viz. (a) *spans not matched by the rule*: unchanged text in $\mathcal{O}$ outside the span matched by the left-hand side regex of the rule can always be linked back to $\mathcal{I}$; and (b) *spans caught by a regex capture group*: capture groups represent the same text in the left- and right-hand sides of a substitution, and so can be linked back to $\mathcal{O}$.[9] Outside these text spans, we can only make definite statements about characterization links at boundary points, which include the start and end of the full string, the start and end of the string

[6] Strictly speaking, there are another two operators, for line-oriented comments and automated versioning of rule files.

[7] For this example, the same effects seemingly could be obtained without iteration (using greatly more complex rules); our actual, non-simplified rules, however, further deal with punctuation marks that can function as prefixes or suffixes, as well as with corner cases like *factor(s)* or *Ca[2+]*. Also in mark-up removal and normalization, we have found it necessary to 'parse' nested structures by means of iterative groups.

[8] If the tokenization process was only concerned with the identification of token boundaries, characterization would be near-trivial.

[9] If capture group references are used out-of-order, however, the per-group linkage is no longer well-defined, and we resort to the maximum-span 'union' of boundary points (see below).

matched by the rule, and the start and end of any capture groups in the rule.

Each character in the string being processed has a start and end position, marking the point before and after the character in the original string. Before processing, the end position would always be one greater than the start position. However, if a rule mapped a string-initial, PTB-style opening double quote (``) to one-character Unicode “, the new first character of the string would have start position 0, but end position 2. In contrast, if there were a rule

```
!wo(n't)    will␣\1        (1)
```

applied to the string *I won't go!*, all characters in the second token of the resulting string (*I will n't go!*) will have start position 2 and end position 4. This demonstrates one of the formal consequences of our design: we have no reason to assign the characters *ill* any start position other than 2.[10] Since explicit character links between each $\mathcal{I}$ and $\mathcal{O}$ will only be established at match or capture group boundaries, any text from the left-hand side of a rule that should appear in $\mathcal{O}$ must be explicitly linked through a capture group reference (rather than merely written out in the right-hand side of the rule). In other words, rule (1) above should be preferred to the following variant (which would result in character start and end offsets of 0 and 5 for *both* output tokens):

```
!won't    will␣n't        (2)
```

During rule application, we keep track of character start and end positions as offsets between a string before and after each rule application (i.e. all pairs $\langle \mathcal{I}, \mathcal{O} \rangle$), and these offsets are eventually traced back to the original string at the time of final tokenization.

6 Quantitative and Qualitative Evaluation

In our own work on preparing various (non-PTB) genres for parsing, we devised a set of REPP rules with the goal of following the PTB conventions. When repeating the experiment of § 3 above using REPP tokenization, we obtained an initial difference in 1505 sentences, with a Levenshtein distance of 3543 (broadly comparable to CoreNLP, if marginally more accurate).

Examining these discrepancies, we revealed some deficiencies in our rules, as well as some peculiarities of the ‘raw’ Wall Street Journal text from the PTB distribution. A little more than 200 mismatches were owed to improper treatment of currency symbols (*AU$*) and decade abbreviations (*’60s*), which led to the refinement of two existing rules. Notable PTB idiosyncrasies (in the sense of deviations from common typography) include ellipses with spaces separating the periods and a fairly large number of possessives (*’s*) being separated from their preceding token. Other aspects of gold-standard PTB tokenization we consider unwarranted ‘damage’ to the input text, such as hallucinating an extra period after `U.S.` and splitting `cannot` (which adds spurious ambiguity). For use cases where the goal were *strict* compliance, for instance in pre-processing inputs for a PTB-derived parser, we added an optional REPP module (of currently half a dozen rules) to cater to these corner cases—in a spirit similar to the CoreNLP mode we used in § 3. With these extra rules, remaining tokenization discrepancies are contained in 603 sentences (just over 1 %), which gives a Levenshtein distance of 1389.

7 Discussion—Conclusion

Compared to the best-performing off-the-shelf system in our earlier experiment (where it is reasonable to assume that PTB data has played at least some role in development), our results eliminate two thirds of the remaining tokenization errors—a more substantial reduction than recent improvements in parsing accuracy against the PTB, for example.

Of the remaining differences, over 350 are concerned with mid-sentence period ambiguity, where at least half of those are instances where a period was separated from an abbreviation in the treebank—a pattern we do not wish to emulate. Some differences in quote disambiguation also remain, often triggered by whitespace on both sides of quote marks in the raw text. The final 200 or so differences stem from manual corrections made during treebanking, and we consider that these cases could not be replicated automatically in any generalizable fashion.

[10] This subtlety will actually be invisible in the final token objects if *will* remains a single token, but if subsequent rules were to split this token further, all its output tokens would have a start position of 2 and an end position of 4. While this example may seem unlikely, we have come across similar scenarios in fine-tuning actual REPP rules.

References

Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics* (pp. 173–180). Ann Arbor, USA.

Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., & Weischedel, R. (2006). Ontonotes. The 90% solution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 57–60). New York City, USA.

Kaplan, R. M. (2005). A method for tokenizing text. Festschrift for Kimmo Koskenniemi on his 60th birthday. In A. Arppe, L. Carlson, K. Lindén, J. Piitulainen, M. Suominen, M. Vainio, H. Westerlund, & A. Yli-Jyrä (Eds.), *Inquiries into words, constraints and contexts* (pp. 55 – 64). Stanford, CA: CSLI Publications.

Levenshtein, V. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physice – Doklady*, *10*, 707–710.

Marcus, M. P., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English. The Penn Treebank. *Computational Linguistics*, *19*, 313 – 330.

Øvrelid, L., Velldal, E., & Oepen, S. (2010). Syntactic scope resolution in uncertainty analysis. In *Proceedings of the 23rd international conference on computational linguistics* (pp. 1379 – 1387). Beijing, China.

Schäfer, U., Kiefer, B., Spurk, C., Steffen, J., & Wang, R. (2011). The ACL Anthology Searchbench. In *Proceedings of the ACL-HLT 2011 system demonstrations* (pp. 7–13). Portland, Oregon, USA.

Tateisi, Y., & Tsujii, J. (2006). *GENIA annotation guidelines for tokenization and POS tagging* (Technical Report # TR-NLP-UT-2006-4). Tokyo, Japan: Tsujii Lab, University of Tokyo.

Waldron, B., Copestake, A., Schäfer, U., & Kiefer, B. (2006). Preprocessing and tokenisation standards in DELPH-IN tools. In *Proceedings of the 5th International Conference on Language Resources and Evaluation* (pp. 2263 – 2268). Genoa, Italy.

Unsupervized Word Segmentation: the case for Mandarin Chinese

Pierre Magistry
Alpage, INRIA & Univ. Paris 7,
175 rue du Chevaleret,
75013 Paris, France
pierre.magistry@inria.fr

Benoît Sagot
Alpage, INRIA & Univ. Paris 7,
175 rue du Chevaleret,
75013 Paris, France
benoit.sagot@inria.fr

Abstract

In this paper, we present an unsupervized segmentation system tested on Mandarin Chinese. Following Harris's Hypothesis in Kempe (1999) and Tanaka-Ishii's (2005) reformulation, we base our work on the Variation of Branching Entropy. We improve on (Jin and Tanaka-Ishii, 2006) by adding normalization and viterbi-decoding. This enable us to remove most of the thresholds and parameters from their model and to reach near state-of-the-art results (Wang et al., 2011) with a simpler system. We provide evaluation on different corpora available from the Segmentation bake-off II (Emerson, 2005) and define a more precise topline for the task using cross-trained supervized system available off-the-shelf (Zhang and Clark, 2010; Zhao and Kit, 2008; Huang and Zhao, 2007)

1 Introduction

The Chinese script has no explicit "word" boundaries. Therefore, tokenization itself, although the very first step of many text processing systems, is a challenging task. Supervized segmentation systems exist but rely on manually segmented corpora, which are often specific to a genre or a domain and use many different segmentation guidelines. In order to deal with a larger variety of genres and domains, or to tackle more theoretic questions about linguistic units, unsupervized segmentation is still an important issue. After a short review of the corresponding literature in Section 2, we discuss the challenging issue of evaluating unsupervized word segmentation systems in Section 3. Section 4 and Section 5 present the core of our system. Finally, in Section 6, we detail and discuss our results.

2 State of the Art

Unsupervized word segmentation systems tend to make use of three different types of information: the cohesion of the resulting units (e.g., Mutual Information, as in (Sproat and Shih, 1990)), the degree of separation between the resulting units (e.g., Accessor Variety, see (Feng et al., 2004)) and the probability of a segmentation given a string (Goldwater et al., 2006; Mochihashi et al., 2009).

A recently published work by Wang et al. (2011) introduce ESA: "Evaluation, Selection, Adjustment." This method combines cohesion and separation measures in a "goodness" metric that is maximized during an iterative process. This work is the current state-of-the-art in unsupervized segmentation of Mandarin Chinese data.

The main drawbacks of ESA are the need to iterate the process on the corpus around 10 times to reach good performance levels and the need to set a parameter that balances the impact of the cohesion measure w.r.t. the separation measure. Empirically, a correlation is found between the parameter and the size of the corpus but this correlation depends on the script used in the corpus (it changes if Latin letters and Arabic numbers are taken into account during preprocessing or not). Moreover, computing this correlation and finding the best value for the parameter (i.e., what the authors call the *proper exponent*) requires a manually segmented training corpus. Therefore, this proper exponent may not be easily available in all situations. However, if we only consider their experiments using settings similar to ours, their results consistently lie around an f-score of 0.80.

An older approach, introduced by Jin and Tanaka-Ishii (2006), solely relies on a separation measure

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 383–387,
Jeju, Republic of Korea, 8-14 July 2012.

that is directly inspired by a linguistic hypothesis formulated by Harris (1955). In Tanaka-Ishii (2005) (following Kempe (1999)) who use Branching Entropy (BE), this hypothesis goes as follows: if sequences produced by human language were random, we would expect the Branching Entropy of a sequence (estimated from the n-grams in a corpus) to decrease as we increase the length of the sequence. Therefore the variation of the branching entropy (VBE) should be negative. When we observe that it is not the case, Harris hypothesizes that we are at a linguistic boundary. Following this hypothesis, (Jin and Tanaka-Ishii, 2006) propose a system that segments when BE is rising or when it reach a certain maximum.

The main drawback of Jin and Tanaka-Ishii (2006) model is that segmentation decisions are taken very locally[1] and do not depend on neighboring cuts. Moreover, this system also also relies on parameters, namely the threshold on the VBE above which the system decides to segment (in their system, this is when VBE$\geq$ 0). In theory, we could expect a decreasing BE and look for a less decreasing value (or on the contrary, rising at least to some extent). A threshold of 0 can be seen as a default value. Finally, Jin and Tanaka-Ishii do not take in account that VBE of n-gram may not be directly comparable to the VBE of m-grams if $m \neq n$. A normalization is needed (as in (Cohen et al., 2002)).

Due to space constraints, we shall not describe here other systems than those by Wang et al. (2011) and Jin and Tanaka-Ishii (2006). A more comprehensive state of the art can be found in (Zhao and Kit, 2008) and (Wang et al., 2011).

In this paper we will show that we can correct the drawbacks of Jin and Tanaka-Ishii (2006) model and reach performances comparable to those of Wang et al. (2011) with as simpler system.

3 Evaluation

In this paper, in order to be comparable with Wang et al. (2011), we evaluate our system against the corpora from the Second International Chinese Word Segmentation Bakeoff (Emerson, 2005). These corpora cover 4 different segmentation guidelines from various origins: Academia Sinica (AS), City-University of Hong-Kong (CITYU), Microsoft Research (MSR) and Peking University (PKU).

[1] Jin (2007) uses self-training with MDL to address this issue.

Evaluating unsupervized systems is a challenge by itself. As an agreement on the exact definition of what a *word* is remains hard to reach, various segmentation guidelines have been proposed and followed for the annotation of different corpora. The evaluation of supervized systems can be achieved on any corpus using any guidelines: when trained on data that follows particular guidelines, the resulting system will follow as well as possible these guidelines, and can be evaluated on data annotated accordingly. However, for unsupervized systems, there is no reason why a system should be closer to one reference than another or even not to lie somewhere in between the different existing guidelines. Huang and Zhao (2007) propose to use cross-training of a supervized segmentation system in order to have an estimation of the consistency between different segmentation guidelines, and therefore an upper bound of what can be expected from an unsupervized system (Zhao and Kit, 2008). The average consistency is found to be as low as 0.85 (f-score). Therefore this figure can be considered as a sensible *topline* for unsupervized systems. The standard *baseline* which consists in segmenting each character leads to a baseline around 0.35 (f-score) — almost half of the tokens in a manually segmented corpus are unigrams.

Per word-length evaluation is also important as units of various lengths tend to have different distributions. We used ZPAR (Zhang and Clark, 2010) on the four corpora from the Second Bakeoff to reproduce Huang and Zhao's (2007) experiments, but also to measure cross-corpus consistency at a per-word-length level. Our overall results are comparable to what Huang and Zhao (2007) report. However, the consistency is quickly falling for longer words: on unigrams, f-scores range from 0.81 to 0.90 (the same as the overall results). We get slightly higher figures on bigrams (0.85–0.92) but much lower on trigrams with only 0.59–0.79. In a segmented Chinese text, most of the tokens are uni- and bigrams but most of the types are bi- and trigrams (as unigrams are often high frequency grammatical words and trigrams the result of more or less productive affixations). Therefore the results of evaluations only based on tokens do not suffer much from poor performances on trigrams even if a large part of the lexicon may be incorrectly processed.

Another issue about the evaluation and comparison of unsupervized systems is to try and remain fair

in terms of preprocessing and prior knowledge given to the systems. For example, Wang et al. (2011) used different levels of preprocessing (which they call "settings"). In their settings 1 and 2, Wang et al. (2011) try not to rely on punctuation and character encoding information (such as distinguishing Latin and Chinese characters). However, they optimize their parameter for each setting. We therefore consider that their system does take into account the level of processing which is performed on Latin characters and Arabic numbers, and therefore "knows" whether to expect such characters or not. In setting 3 they add the knowledge of punctuation as clear boundaries and in setting 4 they preprocess Arabic and Latin and obtain better, more consistent and less questionable results.

As we are more interested in reducing the amount of human labor needed than in achieving by all means fully unsupervized learning, we do not refrain from performing basic and straightforward preprocessing such as detection of punctuation marks, Latin characters and Arabic numbers.[2] Therefore, our experiments rely on settings similar to their settings 3 and 4, and are evaluated against the same corpora.

4 Normalized Variation of Branching Entropy (nVBE)

Our system builds upon Harris's (1955) hypothesis and its reformulation by Kempe (1999) and Tanaka-Ishii (2005). Let us now define formally the notions underlying our system.

Given an n-gram $x_{0..n} = x_{0..1}\, x_{1..2} \ldots x_{n-1..n}$ with a left context $\chi_{\rightarrow}$, we define its *Right Branching Entropy* (RBE) as:

$$\begin{aligned} h_{\rightarrow}(x_{0..n}) &= H(\chi_{\rightarrow} \mid x_{0..n}) \\ &= -\sum_{x \in \chi_{\rightarrow}} P(x \mid x_{0..n}) \log P(x \mid x_{0..n}). \end{aligned}$$

The *Left Branching Entropy* (LBE) is defined in a symmetric way: if we note $\chi_{\leftarrow}$ the right context of $x_{0..n}$, its LBE is defined as:

$$h_{\leftarrow}(x_{0..n}) = H(\chi_{\leftarrow} \mid x_{0..n}).$$

The RBE (resp. LBE) can be considered as $x_{0..n}$'s *Branching Entropy* (BE) when reading from left to right (resp. right to left).

[2]Simple regular expressions could also be considered to deal with unambiguous cases of numbers and dates in Chinese script.

From $h_{\rightarrow}(x_{0..n})$ and $h_{\rightarrow}(x_{0..n-1})$ on the one hand, and from $h_{\leftarrow}(x_{0..n})$ and $h_{\leftarrow}(x_{1..n})$ we estimate the *Variation of Branching Entropy* (VBE) in both directions, defined as follows:

$$\begin{aligned} \delta h_{\rightarrow}(x_{0..n}) &= h_{\rightarrow}(x_{0..n}) - h_{\rightarrow}(x_{0..n-1}) \\ \delta h_{\leftarrow}(x_{0..n}) &= h_{\leftarrow}(x_{0..n}) - h_{\leftarrow}(x_{1..n}). \end{aligned}$$

The VBEs are not directly comparable for strings of different lengths and need to be normalized. In this work, we recenter them around 0 with respect to the length of the string by substracting the mean of the VBEs of the strings of the same length. Writing $\tilde{\delta} h_{\rightarrow}(x)$ and $\tilde{\delta} h_{\leftarrow}(x)$. The normalized VBEs for the string x, or ***nVBEs***, are then defined as follow (we only defined $\tilde{\delta} h_{\leftarrow}(x)$ for clarity reasons): for each length k and each k-gram x such that $len(x) = k$, $\tilde{\delta} h_{\rightarrow}(x) = \delta h_{\rightarrow}(x) - \mu_{\rightarrow,k}$, where $\mu_{\rightarrow,k}$ is the mean of the values of $\delta h_{\rightarrow}(x)$ of all k-grams x.

Note that we use and normalize the variation of branching entropy and not the branching entropy itself. Doing so would break the Harris's hypothesis as we would not expect $\tilde{h}(x_{0..n}) < \tilde{h}(x_{0..n-1})$ in non-boundary situation anymore. Many studies use directly the branching entropy (normalized or not) and report results that are below state-of-the-art systems (Cohen et al., 2002).

5 Decoding algorithm

If we follow Harris's hypothesis and consider complex morphological word structures, we expect a large VBE at the boundaries of interesting units and more unstable variations inside "words." This expectation was confirmed by empirical data visualization. For different lengths of n-grams, we compared the distributions of the VBEs at different positions inside the n-gram and at its boundaries. By plotting density distributions for words vs. non-words, we observed that the VBE at both boundaries were the most discriminative value. Therefore, we decided to take in account the VBE only at the word-candidate boundaries (left and right) and not to consider the inner values. Two interesting consequences of this decision are: first, all $\tilde{\delta} h(x)$ can be precomputed as they do not depend on the context. Second, best segmentation can be computed using dynamic programming.

Since we consider the VBE only at words boundary, we can define for any n-gram w its *autonomy* as $a(x) = \tilde{\delta}_{\leftarrow} h(x) + \tilde{\delta} h_{\rightarrow}(x)$. The more an n-gram is autonomous, the more likely it is to be a word.

With this measure, we can redefine the sentence segmentation problem as the maximization of the autonomy measure of its words. For a character sequence s, if we call $Seg(s)$ the set of all the possible segmentations, then we are looking for:

$$\arg\max_{W \in Seg(s)} \sum_{w_i \in W} a(w_i) \cdot len(w_i),$$

where W is the segmentation corresponding to the sequence of words $w_0 w_1 \dots w_m$, and $len(w_i)$ is the length of a word w_i used here to be able to compare segmentations resulting in a different number of words. This best segmentation can be computed easily using dynamic programming.

6 Results and discussion

We tested our system against the data from the 4 corpora of the Second Bakeoff, in both settings 3 and 4, as described in Section 3. Overall results are given in Table 1 and per-word-length results in Table 2.

Our results (nVBE) show significant improvements over Jin's (2006) strategy (VBE $>$ 0) and are closely competing with ESA. But contrarily to ESA (Wang et al., 2011), it does not require multiple iterations on the corpus and it does not rely on any parameters. This shows that we can rely solely on a separation measure and get high segmentation scores. When maximized over a sentence, this measure captures at least in part what can be modeled by a cohesion measure without the need for fine-tuning the balance between the two.

The evolution of the results w.r.t. word length is consistent with the supervized cross-evaluation results of the various segmentation guidelines as performed in Section 3.

Due to space constraints, we cannot detail here a qualitative analysis of the results. We can simply mention that the errors we observed are consistent with previous systems based on Harris's hypothesis (see (Magistry and Sagot, 2011) and Jin (2007) for a longer discussion). Many errors are related to dates and Chinese numbers. This could and should be dealt with during preprocessing. Other errors often involve frequent grammatical morphemes or productive affixes. These errors are often interesting for linguists and could be studied as such and/or corrected in a post-processing stage that would introduce linguistic knowledge. Indeed, unlike content words, grammatical morphemes belongs to closed classes, therefore introducing this linguistic knowledge into the system may be of great help without requiring to much human effort. A sensible way to go in that direction would be to let unsupervized system deal with open classes and process closed classes with a symbolic or supervized module.

System	AS	CITYU	PKU	MSR
Setting 3				
ESA worst	0.729	0.795	0.781	0.768
ESA best	0.782	0.816	0.795	0.802
nVBE	0.758	0.775	0.781	0.798
Setting 4				
VBE >0	0.63	0.640	0.703	0.713
ESA worst	0.732	0.809	0.784	0.784
ESA best	0.786	0.829	0.800	0.818
nVBE	0.766	0.767	0.800	0.813

Table 1: Evaluation on the Second Bakeoff data with Wang et al.'s (2011) settings. "Worst" and "best" give the range of the reported results with differents values of the parameter in Wang et al.'s system. VBE > 0 correspond to a cut whenever BE is raising. nVBE corresponds to our proposal, based on normalized VBE with maximization at word boundaries. Recall that the topline is around 0.85

Corpus	overall	unigrams	bigrams	trigrams
AS	0.766	0.741	0.828	0.494
CITYU	0.767	0.739	0.834	0.555
PKU	0.800	0.789	0.855	0.451
MSR	0.813	0.823	0.856	0.482

Table 2: Per word-length details of our results with our nVBE algorithm and setting 4. Recall that the toplines are respectively 0.85, 0.81, 0.85 and 0.59 (see Section 3)

One can also observe that our system performs better on PKU and MSR corpora. As PKU is the smallest corpus and AS the biggest, size alone cannot explain this result. However, PKU is more consistent in genre as it contains only articles from the People's Daily. On the other end, AS is a balanced corpus with a greater variety in many aspects. CITYU Corpus is almost as small as PKU but contains articles from newspapers of various Mandarin Chinese speaking communities where great variation is to be expected. This suggest that consistency of the input data is as important as the amount of data. This hypothesis has to be confirmed in futur studies. If it is, automatic clustering of the input data may be an important pre-processing step for this kind of systems.

References

Paul Cohen, Brent Heeringa, and Niall Adams. 2002. An unsupervised algorithm for segmenting categorical timeseries into episodes. *Pattern Detection and Discovery*, page 117–133.

Thomas Emerson. 2005. The second international chinese word segmentation bakeoff. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, volume 133.

Haodi Feng, Kang Chen, Xiaotie Deng, and Weiming Zheng. 2004. Accessor variety criteria for Chinese word extraction. *Computational Linguistics*, 30(1):75–93.

Sharon Goldwater, Thomas L. Griffiths, and Mark Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, page 673–680.

Zellig S. Harris. 1955. From phoneme to morpheme. *Language*, 31(2):190–222.

Changning. Huang and Hai Zhao. 2007. 中文分词十年回顾 (Chinese word segmentation: A decade review). *Journal of Chinese Information Processing*, 21(3):8–20.

Zhihui Jin and Kumiko Tanaka-Ishii. 2006. Unsupervised segmentation of Chinese text by use of branching entropy. In *Proceedings of the COLING/ACL on Main conference poster sessions*, page 428–435.

Zhihui Jin. 2007. *A Study On Unsupervised Segmentation Of Text Using Contextual Complexity*. Ph.D. thesis, University of Tokyo.

André Kempe. 1999. Experiments in unsupervised entropy-based corpus segmentation. In *Workshop of EACL in Computational Natural Language Learning*, page 7–13.

Pierre Magistry and Benoît Sagot. 2011. Segmentation et induction de lexique non-supervisées du mandarin. In *TALN'2011 - Traitement Automatique des Langues Naturelles*, Montpellier, France, June. ATALA.

Daichi Mochihashi, Takeshi. Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. 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 100–108.

Richard W. Sproat and Chilin Shih. 1990. A statistical method for finding word boundaries in Chinese text. *Computer Processing of Chinese and Oriental Languages*, 4(4):336–351.

Kumiko Tanaka-Ishii. 2005. Entropy as an indicator of context boundaries: An experiment using a web search engine. In *IJCNLP*, page 93–105.

Hanshi Wang, Jian Zhu, Shiping Tang, and Xiaozhong Fan. 2011. A new unsupervised approach to word segmentation. *Computational Linguistics*, 37(3):421–454.

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*, page 843–852.

Hai Zhao and Chunyu Kit. 2008. An empirical comparison of goodness measures for unsupervised Chinese word segmentation with a unified framework. In *The Third International Joint Conference on Natural Language Processing (IJCNLP2008), Hyderabad, India*.

Grammar Error Correction Using Pseudo-Error Sentences and Domain Adaptation

Kenji Imamura, Kuniko Saito, Kugatsu Sadamitsu, and **Hitoshi Nishikawa**
NTT Cyber Space Laboratories, NTT Corporation
1-1 Hikari-no-oka, Yokosuka, 239-0847, Japan
{imamura.kenji, saito.kuniko, sadamitsu.kugatsu, nishikawa.hitoshi}@lab.ntt.co.jp

Abstract

This paper presents grammar error correction for Japanese particles that uses discriminative sequence conversion, which corrects erroneous particles by substitution, insertion, and deletion. The error correction task is hindered by the difficulty of collecting large error corpora. We tackle this problem by using pseudo-error sentences generated automatically. Furthermore, we apply domain adaptation, the pseudo-error sentences are from the source domain, and the real-error sentences are from the target domain. Experiments show that stable improvement is achieved by using domain adaptation.

1 Introduction

Case marks of a sentence are represented by postpositional particles in Japanese. Incorrect usage of the particles causes serious communication errors because the cases become unclear. For example, in the following sentence, it is unclear what must be deleted.

mail **o** *todoi tara sakujo onegai-shi-masu*
mail **ACC.** arrive when delete please
"When ϕ has arrived an e-mail, please delete it."

If the accusative particle *o* is replaced by a nominative one *ga*, it becomes clear that the writer wants to delete the e-mail ("When the e-mail has arrived, please delete it."). Such particle errors frequently occur in sentences written by non-native Japanese speakers.

This paper presents a method that can automatically correct Japanese particle errors. This task corresponds to preposition/article error correction in English. For English error correction, many studies employ classifiers, which select the appropriate prepositions/articles, by restricting the error types to articles and frequent prepositions (Gamon, 2010; Han et al., 2010; Rozovskaya and Roth, 2011).

On the contrary, Mizumoto et al. (2011) proposed translator-based error correction. This approach can handle all error types by converting the learner's sentences into the correct ones. Although the target of this paper is particle error, we employ a similar approach based on sequence conversion (Imamura et al., 2011) since this offers excellent scalability.

The conversion approach requires pairs of the learner's and the correct sentences. However, collecting a sufficient number of pairs is expensive. To avoid this problem, we use additional corpus consisting of pseudo-error sentences automatically generated from correct sentences that mimic the real-errors (Rozovskaya and Roth, 2010b). Furthermore, we apply a domain adaptation technique that regards the pseudo-errors and the real-errors as the source and the target domain, respectively, so that the pseudo-errors better match the real-errors.

2 Error Correction by Discriminative Sequence Conversion

We start by describing discriminative sequence conversion. Our error correction method converts the learner's word sequences into the correct sequences. Our method is similar to phrase-based statistical machine translation (PBSMT), but there are three differences; 1) it adopts the conditional random fields, 2) it allows insertion and deletion, and 3) binary and real features are combined. Unlike the classification

Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics, pages 388–392,
Jeju, Republic of Korea, 8-14 July 2012.

Incorrect Particle	Correct Particle	Note
ϕ	no/POSS.	INS
ϕ	o/ACC.	INS
ga/NOM.	o/ACC.	SUB
o/ACC.	ni/DAT.	SUB
o/ACC.	ga/NOM.	SUB
wa/TOP.	o/ACC.	SUB
no/POSS.	ϕ	DEL
:	:	

Table 1: Example of Phrase Table (partial)

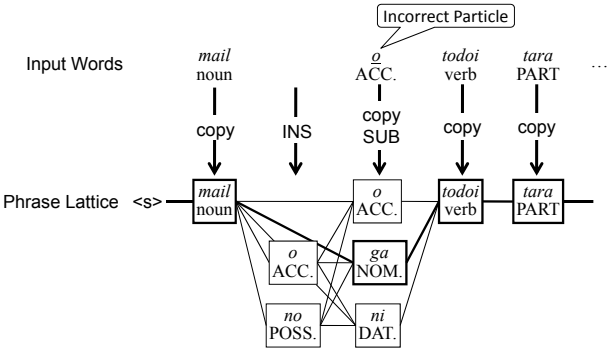


Figure 1: Example of Phrase Lattice

approach, the conversion approach can correct multiple errors of all types in a sentence.

2.1 Basic Procedure

We apply the morpheme conversion approach that converts the results of a speech recognizer into word sequences for language analyzer processing (Imamura et al., 2011). It corrects particle errors in the input sentences as follows.

- First, all modification candidates are obtained by referring to a phrase table. This table, called the confusion set (Rozovskaya and Roth, 2010a) in the error correction task, stores pairs of incorrect and correct particles (Table 1). The candidates are packed into a lattice structure, called the phrase lattice (Figure 1). To deal with unchanged words, it also copies the input words and inserts them into the phrase lattice.
- Next, the best phrase sequence in the phrase lattice is identified based on the conditional random fields (CRFs (Lafferty et al., 2001)). The Viterbi algorithm is applied to the decoding because error correction does not change the word order.
- While training, word alignment is carried out by dynamic programming matching. From the alignment results, the phrase table is constructed by acquiring particle errors, and the CRF models are trained using the alignment results as supervised data.

2.2 Insertion / Deletion

Since an insertion can be regarded as replacing an empty word with an actual word, and deletion is the replacement of an actual word with an empty one, we treat these operations as substitution without distinction while learning/applying the CRF models.

However, insertion is a high cost operation because it may occur at any location and can cause lattice size to explode. To avoid this problem, we permit insertion only immediately after nouns.

2.3 Features

In this paper, we use mapping features and link features. The former measure the correspondence between input and output words (similar to the translation models of PBSMT). The latter measure the fluency of the output word sequence (similar to language models).

The mapping features are all binary. The focusing phrase and its two surrounding words of the input are regarded as the window. The mapping features are defined as the pairs of the output phrase and 1-, 2-, and 3-grams in the window.

The link features are important for the error correction task because the system has to judge output correctness. Fortunately, CRF, which is a kind of discriminative model, can handle features that depend on each other; we mix two types of features as follows and optimize their weights in the CRF framework.

- **N-gram features:** N-grams of the output words, from 1 to 3, are used as binary features. These are obtained from a training corpus (paired sentences). Since the feature weights are optimized considering the entire feature space, fine-tuning can be achieved. The accuracy becomes almost perfect on the training corpus.
- **Language model probability:** This is a logarithmic value (real value) of the n-gram probability of the output word sequence. One feature weight is assigned. The n-gram language model can be

constructed from a large sentence set because it does not need the learner's sentences.

Incorporating binary and real features yields a rough approximation of generative models in semi-supervised CRFs (Suzuki and Isozaki, 2008). It can appropriately correct new sentences while maintaining high accuracy on the training corpus.

3 Pseudo-error Sentences and Domain Adaptation

The error corrector described in Section 2 requires paired sentences. However, it is expensive to collect them. We resolve this problem by using pseudo-error sentences and domain adaptation.

3.1 Pseudo-Error Generation

Correct sentences, which are halves of the paired sentences, can be easily acquired from corpora such as newspaper articles. Pseudo-errors are generated from them by the substitution, insertion, and deletion functions according to the desired error patterns.

We utilize the method of Rozovskaya and Roth (2010b). Namely, when particles appear in the correct sentence, they are replaced by incorrect ones in a probabilistic manner by applying the phrase table (which stores the error patterns) in the opposite direction. The error generation probabilities are relative frequencies on the training corpus. The models are learnt using both the training corpus and the pseudo-error sentences.

3.2 Adaptation by Feature Augmentation

Although the error generation probabilities are computed from the real-error corpus, the error distribution that results may be inappropriate. To better fit the pseudo-errors to the real-errors, we apply a domain adaptation technique. Namely, we regard the pseudo-error corpus as the source domain and the real-error corpus as the target domain, and models are learnt that fit the target domain.

In this paper, we use Daume (2007)'s feature augmentation method for the domain adaptation, which eliminates the need to change the learning algorithm. This method regards the models for the source domain as the prior distribution and learns the models for the target domain.

	Feature Space: Common	Source	Target
Source Data	$\boldsymbol{D_s}$	$\boldsymbol{D_s}$	$\mathbf{0}$
Target Data	$\boldsymbol{D_t}$	$\mathbf{0}$	$\boldsymbol{D_t}$

Figure 2: Feature Augmentation

We briefly review feature augmentation. The feature space is segmented into three parts: common, source, and target. The features extracted from the source domain data are deployed to the common and the source spaces, and those from the target domain data are deployed to the common and the target spaces. Namely, the feature space is tripled (Figure 2).

The parameter estimation is carried out in the usual way on the above feature space. Consequently, the weights of the common features are emphasized if the features are consistent between the source and the target. With regard to domain dependent features, the weights in the source or the target space are emphasized.

Error correction uses only the features in the common and target spaces. The error distribution approaches that of the real-errors because the weights of features are optimized to the target domain. In addition, it becomes robust against new sentences because the common features acquired from the source domain can be used even when they do not appear in the target domain.

4 Experiments

4.1 Experimental Settings

Real-error Corpus: We collected learner's sentences written by Chinese native speakers. The sentences were created from English Linux manuals and figures, and Japanese native speakers revised them. From these sentences, only particle errors were retained; the other errors were corrected. As a result, we obtained 2,770 paired sentences. The number of incorrect particles was 1,087 (8.0%) of 13,534. Note that most particles did not need to be revised. The number of pair types of incorrect particles and their correct ones was 132.

Language Model: It was constructed from Japanese Wikipedia articles about computers and

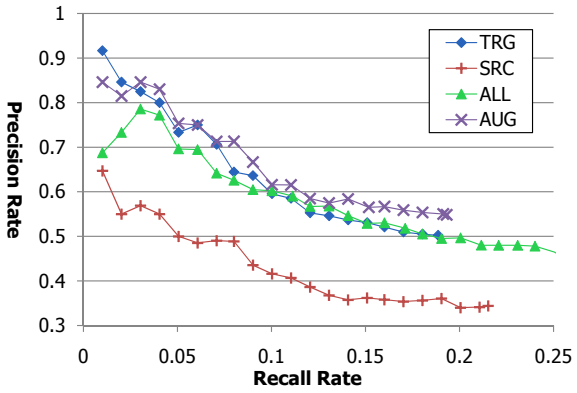


Figure 3: Recall/Precision Curve (Error Generation Magnification is 1.0)

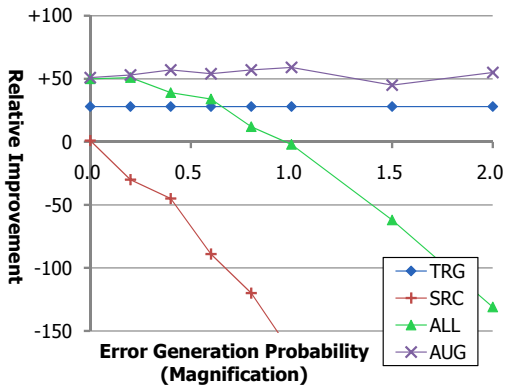


Figure 4: Relative Improvement among Error Generation Probabilities

Japanese Linux manuals, 527,151 sentences in total. SRILM (Stolcke et al., 2011) was used to train a trigram model.

Pseudo-error Corpus: The pseudo-errors were generated using 10,000 sentences randomly selected from the corpus for the language model. The magnification of the error generation probabilities was changed from 0.0 (i.e., no errors) to 2.0 (the relative frequency in the real-error corpus was taken as 1.0).

Evaluation Metrics: Five-fold cross-validation on the real-error corpus was used. We used two metrics: 1) Precision and recall rates of the error correction by the systems, and 2) Relative improvement, the number of differences between improved and degraded particles in the output sentences (no changes were ignored). This is a practical metric because it denotes the number of particles that human rewriters do not need to revise after the system correction.

4.2 Results

Figure 3 plots the precision/recall curves for the following four combinations of training corpora and method.

- **TRG:** The models were trained using only the real-error corpus (baseline).
- **SRC:** Trained using only the pseudo-error corpus.
- **ALL:** Trained using the real-error and pseudo-error corpora by simply adding them.
- **AUG:**
  The proposed method. The feature augmentation was realized by regarding the pseudo-errors as the source domain and the real-errors as the target domain.

The SRC case, which uses only the pseudo-error sentences, did not match the precision of TRG. The ALL case matched the precision of TRG at high recall rates. AUG, the proposed method, achieved higher precision than TRG at high recall rates. At the recall rate of 18%, the precision rate of AUG was 55.4%; in contrast, that of TRG was 50.5%. Feature augmentation effectively leverages the pseudo-errors for error correction.

Figure 4 shows the relative improvement of each method according to the error generation probabilities. In this experiment, ALL achieved higher improvement than TRG at error generation probabilities ranging from 0.0 to 0.6. Although the improvements were high, we have to control the error generation probability because the improvements in the SRC case fell as the magnification was raised. On the other hand, AUG achieved stable improvement regardless of the error generation probability. We can conclude that domain adaptation to the pseudo-error sentences is the preferred approach.

5 Conclusions

This paper presented an error correction method of Japanese particles that uses pseudo-error generation. We applied domain adaptation in which the pseudo-errors are regarded as the source domain and the real-errors as the target domain. In our experiments, domain adaptation achieved stable improvement in system performance regardless of the error generation probability.

References

Hal Daume, III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007)*, pages 256–263, Prague, Czech Republic.

Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2010)*, pages 163–171, Los Angeles, California.

Na-Rae Han, Joel Tetreault, Soo-Hwa Lee, and Jin-Young Ha. 2010. Using an error-annotated learner corpus to develop an ESL/EFL error correction system. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta.

Kenji Imamura, Tomoko Izumi, Kugatsu Sadamitsu, Kuniko Saito, Satoshi Kobashikawa, and Hirokazu Masataki. 2011. Morpheme conversion for connecting speech recognizer and language analyzers in unsegmented languages. In *Proceedings of Interspeech 2011*, pages 1405–1408, Florence, Italy.

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-2001)*, pages 282–289, Williamstown, Massachusetts.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In *Proceedings of 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 147–155, Chiang Mai, Thailand.

Alla Rozovskaya and Dan Roth. 2010a. Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, pages 961–970, Cambridge, Massachusetts.

Alla Rozovskaya and Dan Roth. 2010b. Training paradigms for correcting errors in grammar and usage. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT 2010)*, pages 154–162, Los Angeles, California.

Alla Rozovskaya and Dan Roth. 2011. Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Techologies (ACL-HLT 2011)*, pages 924–933, Portland, Oregon.

Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. SRILM at sixteen: Update and outlook. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU 2011)*, Waikoloa, Hawaii.

Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, pages 665–673, Columbus, Ohio.

Author Index